Hands-On Tutorial SAP Predictive Analytics, Automated Mode: Multiple Time Series

This hands-on guide shows how to forecast multiple time series at once with SAP Predictive Analytics, Automated Mode. Use the scripting functionality to produce forecasts en masse.

The data used in this guide is publicly available so that the reader can follow hands-on and carry out the same analysis.

Please note that this guide is giving a high-level introductory overview and only shows a small fraction of the available functionality.

October 2015

Andreas Forster
Predictive Presales Expert
SAP Switzerland
andreas.forster@sap.com

TABLE OF CONTENTS

INTRODUCTION

SAP Predictive Analytics focusses strongly on automating common predictive requirements to increase the productivity of Data Scientists or Data Analysts. A very comprehensive framework, which follows rigid statistical procedures, shields the user from the underlying complexity.

As such, SAP Predictive Analytics can automated for instance

- Classification
- Regression
- Clustering
- Network Analysis
- Time Series Analysis

These tasks can be all carried out easily in the tool's Graphical User Interface. With only a few clicks for example, a single time series can be forecasted.

This document explains how to apply the same forecasting mechanism to multiple time series at once through a scripting interface. You do not have to create the script from scratch. SAP Predictive Analytics can provide you with a script that forecasts one single time series. This then just needs to be customized to handle as many time series as needed.

The high degree of automated forecasting is shown by forecasting the numbers of road accidents with personal injury in Switzerland. The forecasts will be based on historic data shared by the Swiss Federal Statistical Office[1]. This historic data is broken down by month and the accident location (City, Countryside or Motorway). By the end of this tutorial we will have created individual monthly forecasts for each location.

The datasets AccidentsCityOnly.csv and *AccidentsMultipleLocations.csv* used in this tutorial are available together with this document on GitHub[2]. For your reference, you can also find there the final scripts produced in this document. The tutorial uses the folder C:\Forecasting to store the datasets and to save the forecasts in, feel free to select a different folder if you prefer.

Please note that the datasets are very simple examples, each month is described only by a single value. Additional detail (ie daily values) or additional columns (such as weather information) would improve the forecast.

If you would like to try out a single forecast with the Graphical User Interface using numerous predictor variables, please follow the "Time Series" user guide[3], which includes an example to forecast a company's cash flow.

My thanks go to Mikal Netteberg who explained to me the concept and details of producing such time series forecasts though script.

[1] http://www.bfs.admin.ch/bfs/portal/en/index.html
[2] https://github.com/AndreasForster/Predictive/raw/master/AutomatedTimeSeriesBatchTutorial.zip
[3] http://help.sap.com/pa

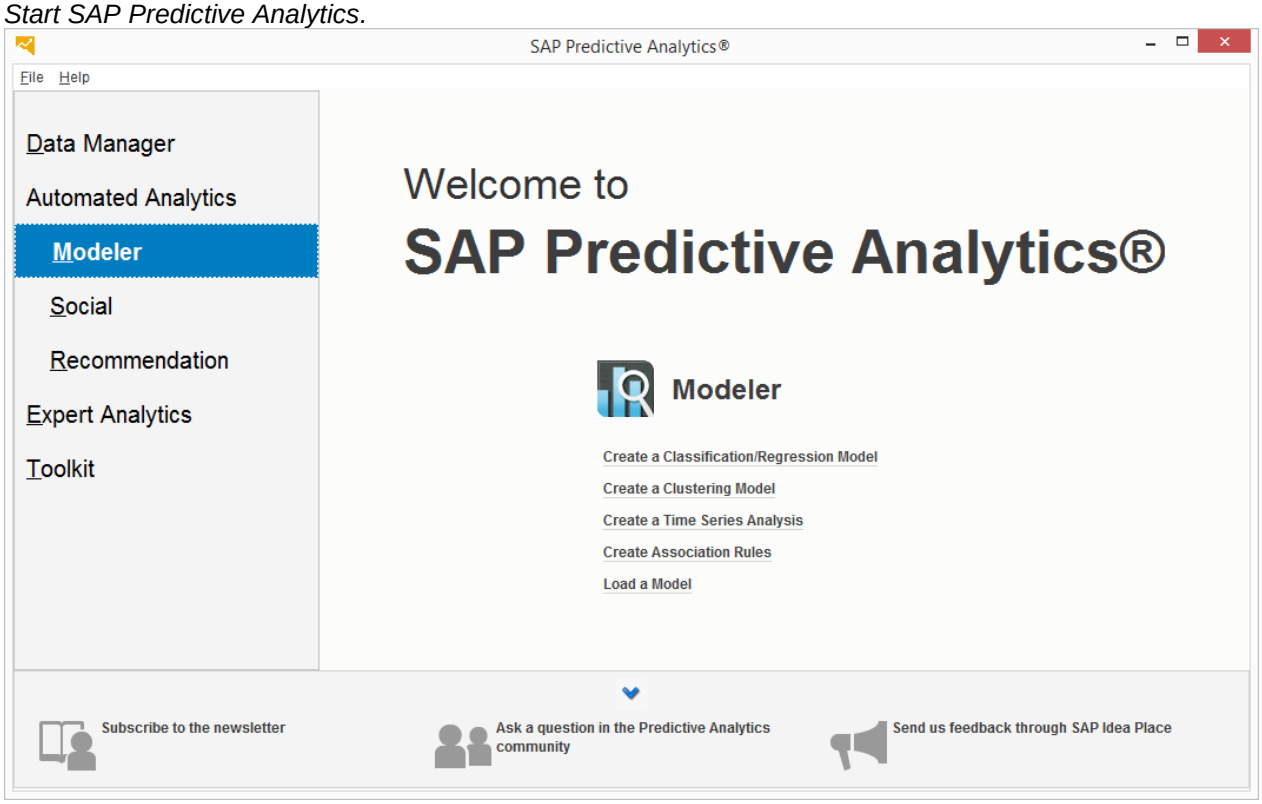
HANDS-ON IMPLEMENTATION

Forecasting one time series manually

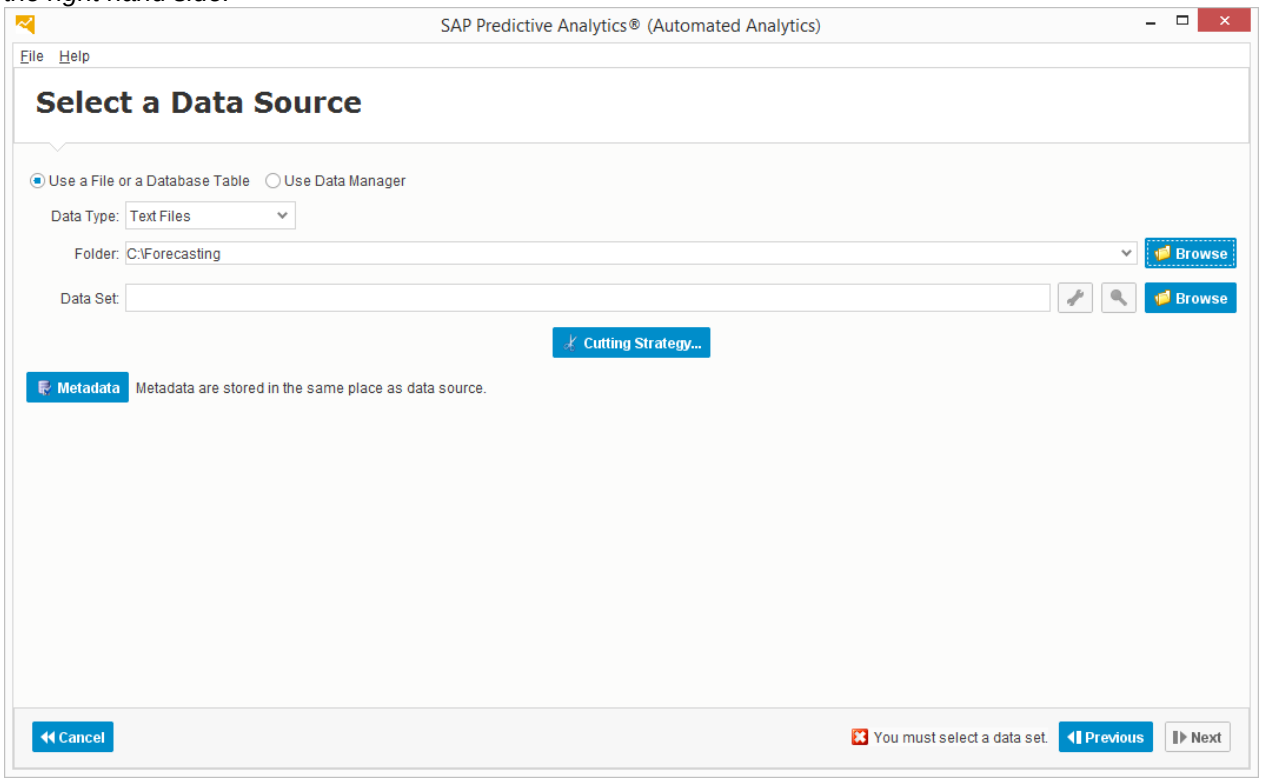
To become comfortable with producing multiple forecasts through the scripting language, it is important to be familiar with forecasting a single time series in the graphical interface of SAP Predictive Analytics.

This document will not elaborate deeply on this part, but as a brief introduction (or a recap if you have used it before) you can follow a simple example. We forecast just the accident numbers happening in Cities

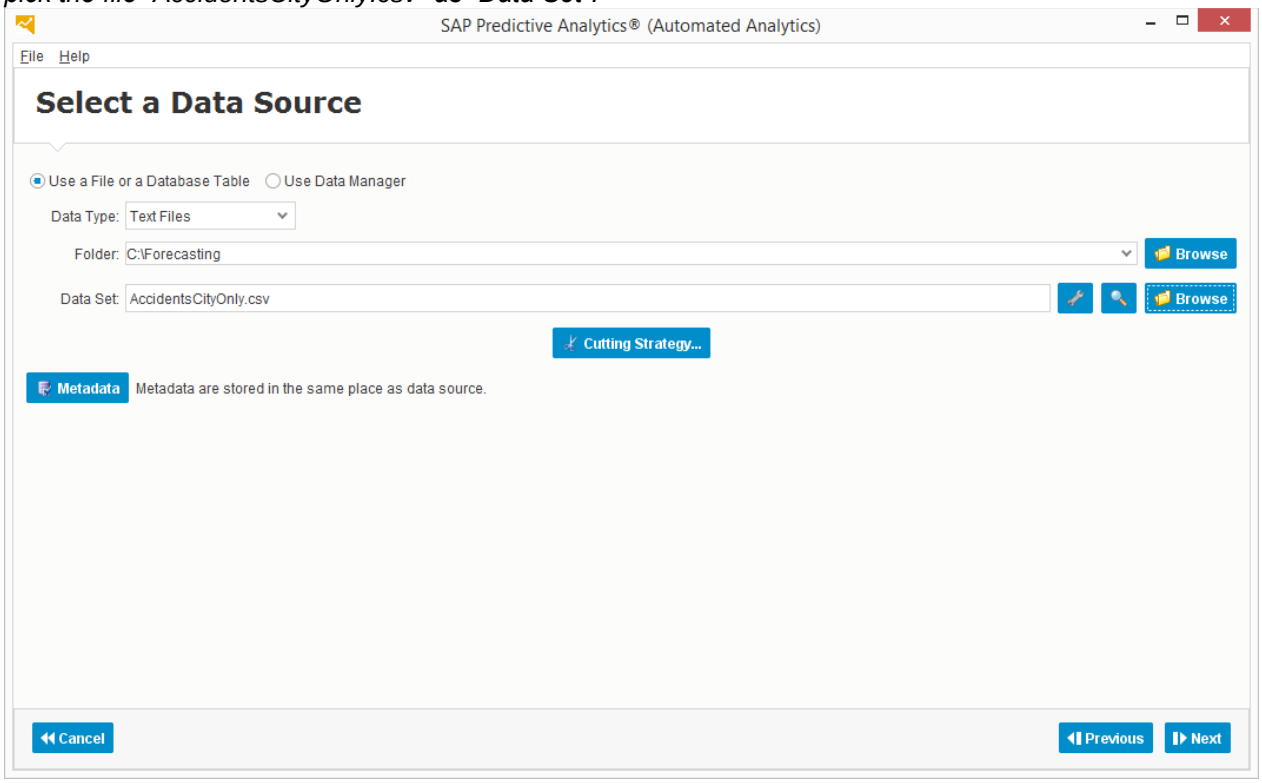
Start SAP Predictive Analytics.



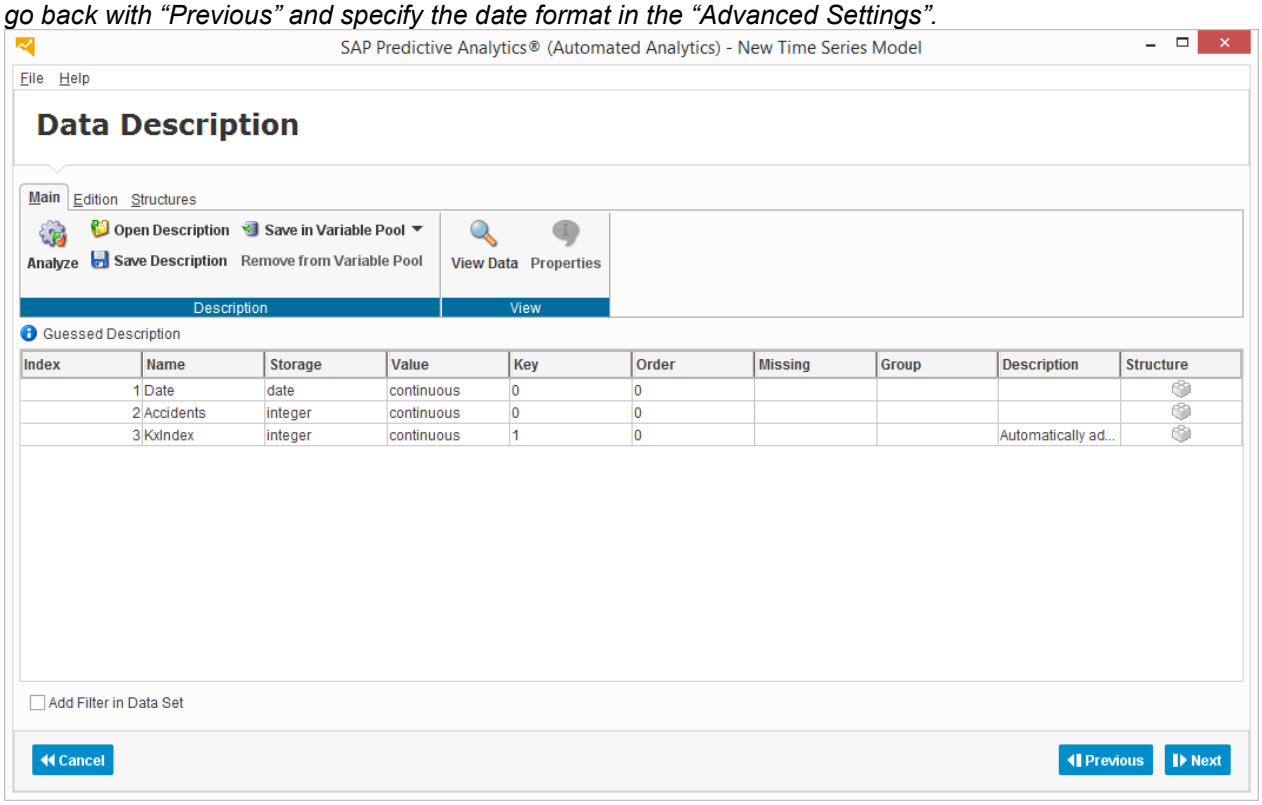
Select the "Modeler" section under "Automated Analytics" and click into "Create a Time Series Analysis" on the right hand side.



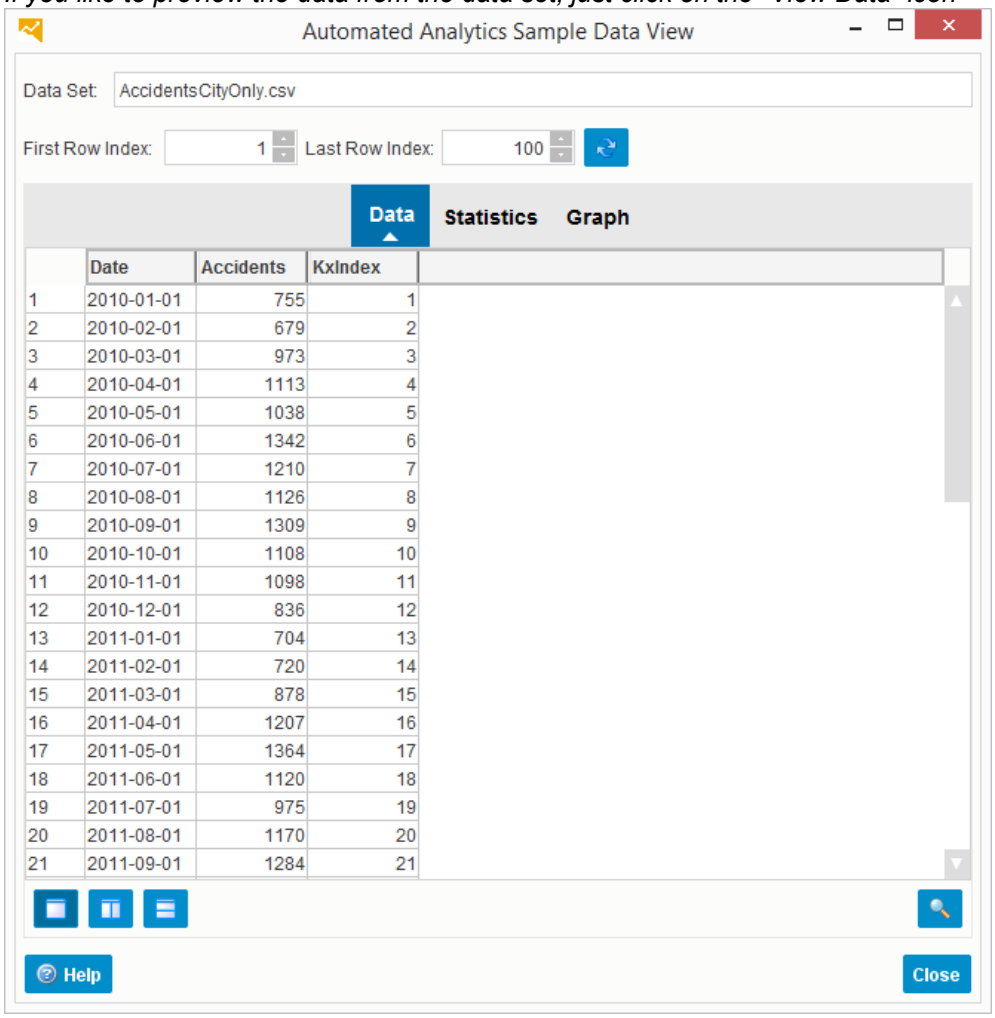
Select our data file by using the following settings:
Ensure "Use a File or a Database Table" is selected. Set "Data Type" to "Text Files". Specify the folder and pick the file" AccidentsCityOnly.csv" as "Data Set".



Click "Next". In the "Data Description" window, click on "Analyze" for SAP Predictive Analytics to guess the data structure. It is very important that the date column has the Storage-type of "date". If this is not the case, go back with "Previous" and specify the date format in the "Advanced Settings".



If you like to preview the data from the data set, just click on the "View Data" icon

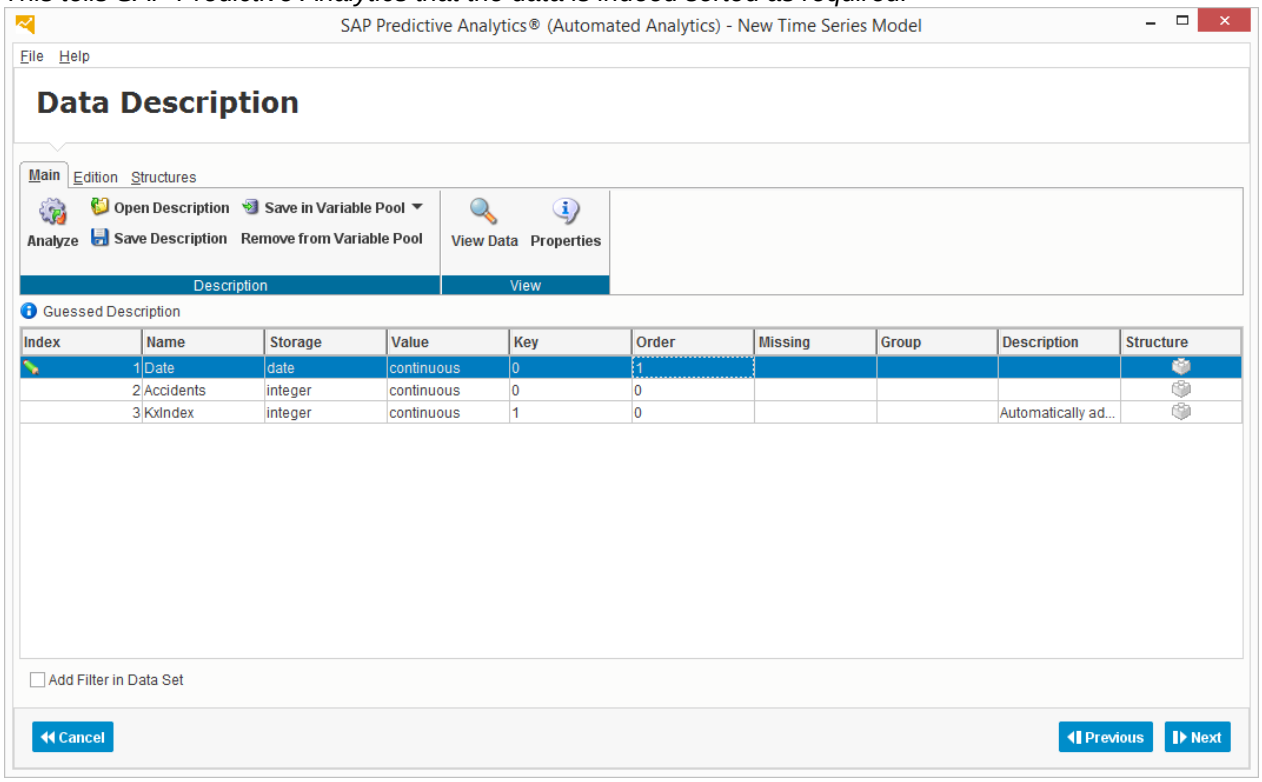


	Date	Accidents	KxIndex
1	2010-01-01	755	1
2	2010-02-01	679	2
3	2010-03-01	973	3
4	2010-04-01	1113	4
5	2010-05-01	1038	5
6	2010-06-01	1342	6
7	2010-07-01	1210	7
8	2010-08-01	1126	8
9	2010-09-01	1309	9
10	2010-10-01	1108	10
11	2010-11-01	1098	11
12	2010-12-01	836	12
13	2011-01-01	704	13
14	2011-02-01	720	14
15	2011-03-01	878	15
16	2011-04-01	1207	16
17	2011-05-01	1364	17
18	2011-06-01	1120	18
19	2011-07-01	975	19
20	2011-08-01	1170	20
21	2011-09-01	1284	21

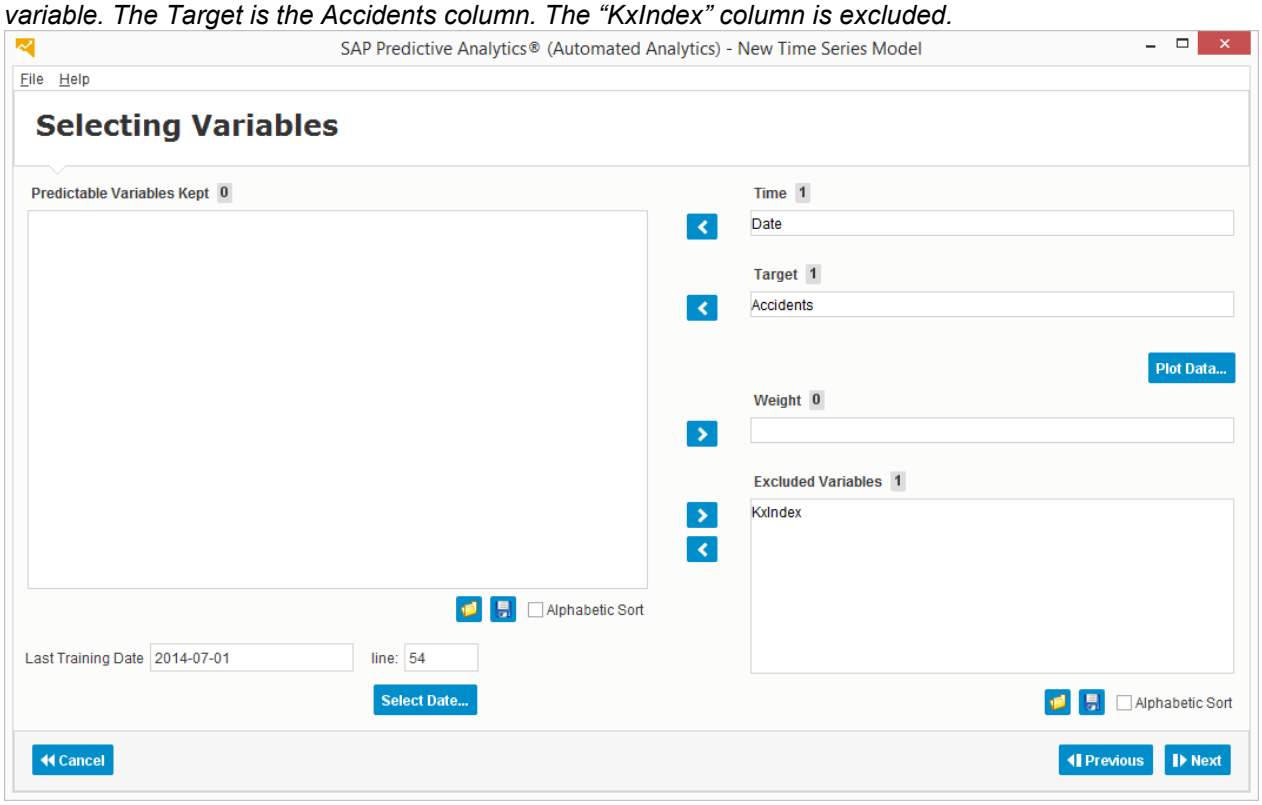
You notice that the data is sorted descending on the date column. This is a requirement for the time series forecast. When working with your own data you need to ensure your data is also sorted beginning with the oldest time stamps. The "KxIndex" column has been added by SAP Predictive Analytics.

Scroll the records down and you see we have historical data from January 2010 to June 2014. For each month in that time frame the actual number of accidents is known. However, the data set continues, there are also rows for the months up to December 2014 but the column with the accident numbers is empty. Having the dates you want to forecast in the dataset is a requirement in case you have additional predictor variables. In that case, these predictor variables must also be filled for the dates you want to forecast. So the data structure is ready for additional predictor variable, such as weather. In this tutorial we keep the structure as it is though.

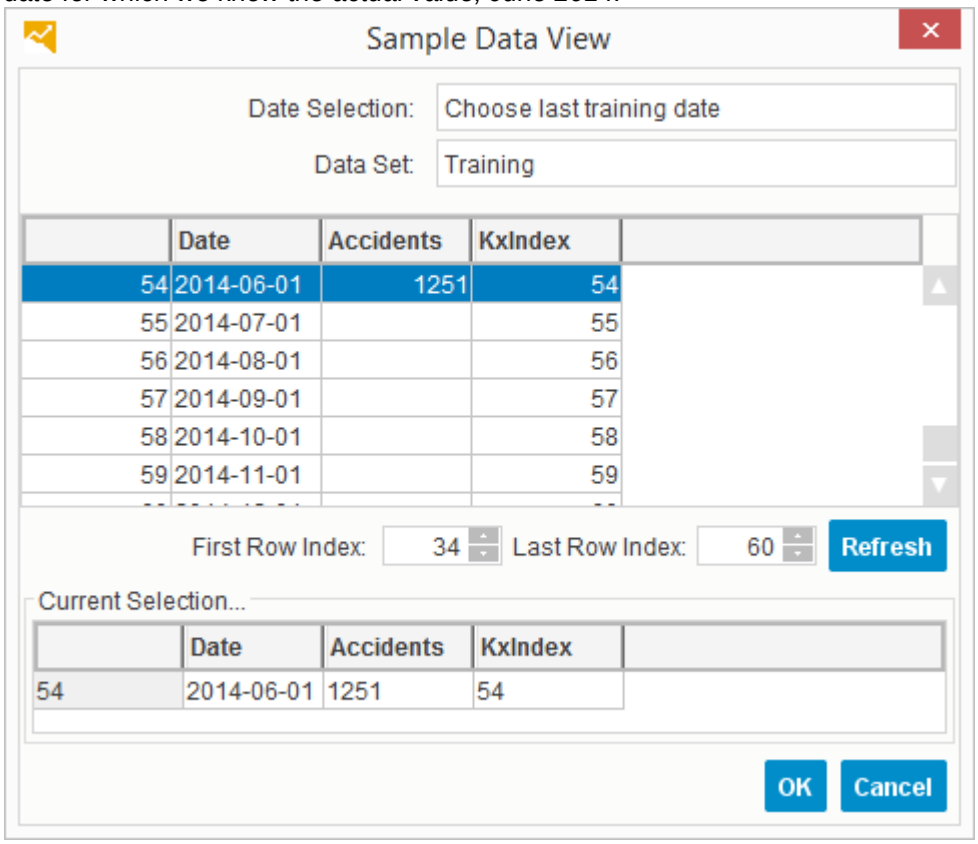
Close the Window to continue. Back in the "Data Description" window, set the Order of the date column to 1. This tells SAP Predictive Analytics that the data is indeed sorted as required.



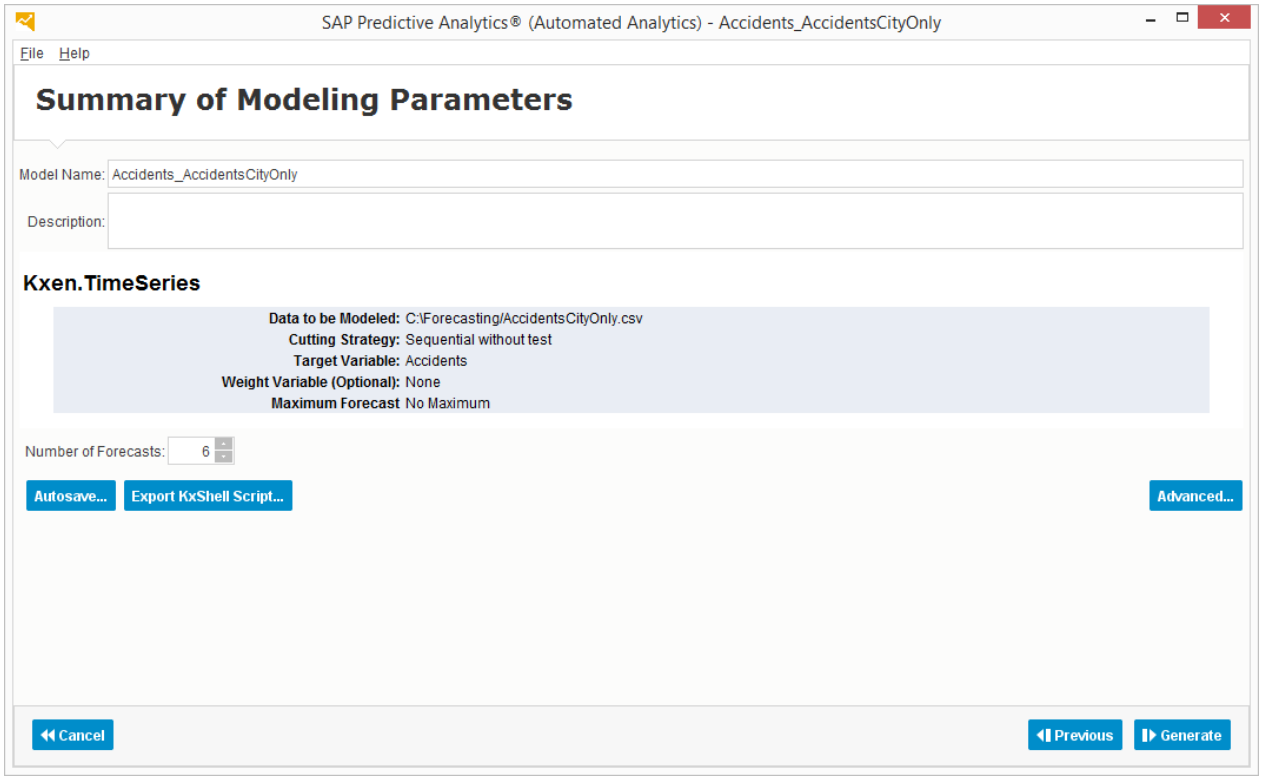
Click Next and ensure the "Selecting Variables" parameters are as follows. Set the "Time" to the Date variable. The Target is the Accidents column. The "KxIndex" column is excluded.



Click "Select Date…" and you see that the last data point to train the model has been set to the most recent date for which we know the actual value, June 2014.



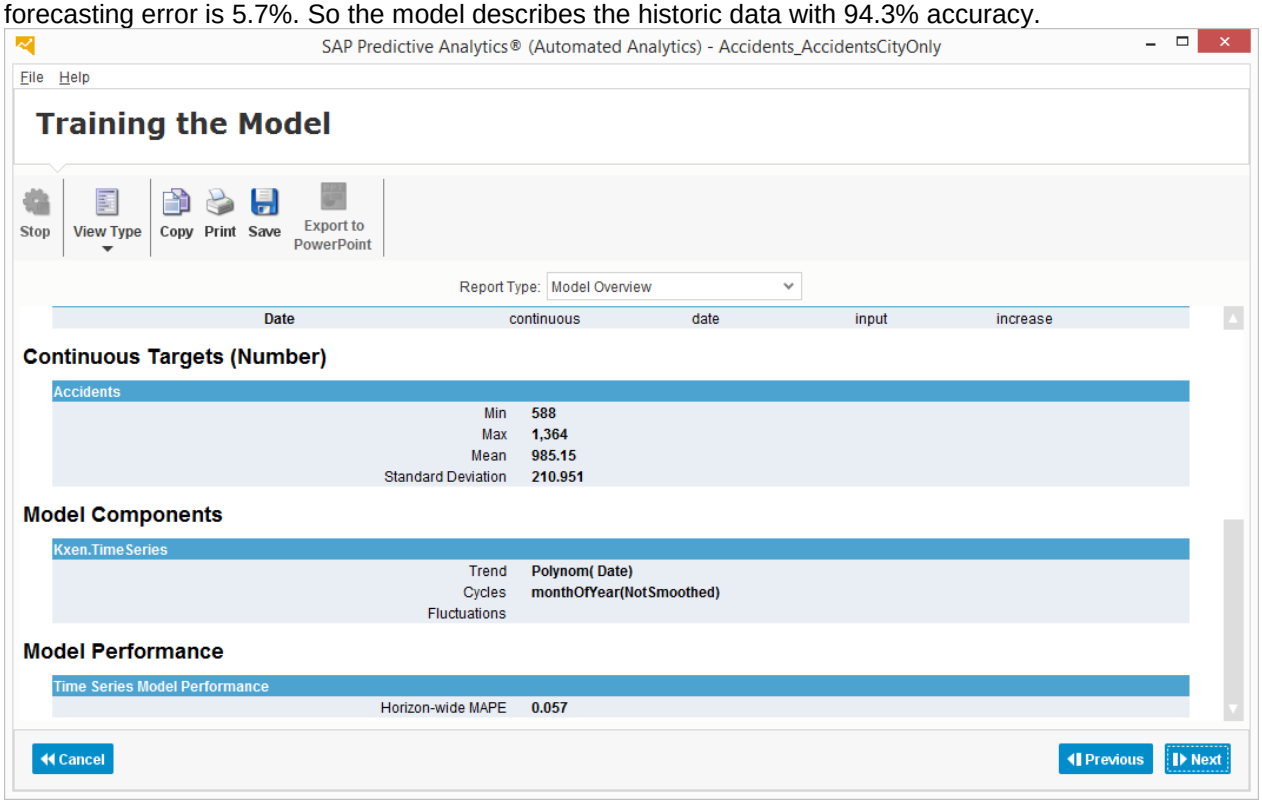
Close this window with "OK" and hit "Next". Set the "Number of Forecasts" to 6.



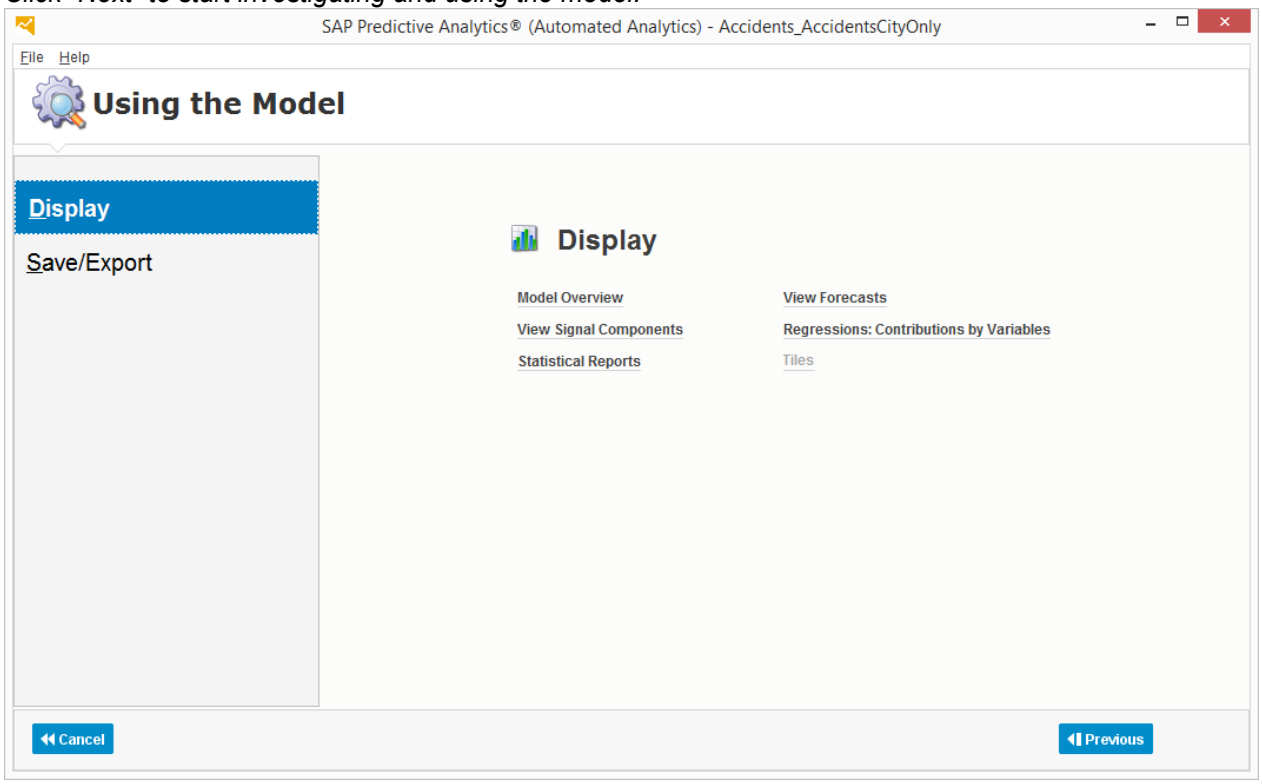
Continue with "Generate" to train the model that describes the historic time series. When completed, scroll down on the "Model Overview" page to see the MAPE value, which describes how good the model is fitted to the training data. MAPE stands for Mean Absolute Percentage Error.

Quoting the help file: "The MAPE value is the average of the sum of the absolute values of the percentage errors. It measures the accuracy of the model's forecasts and indicates how much the forecasts differ from the real signal value."

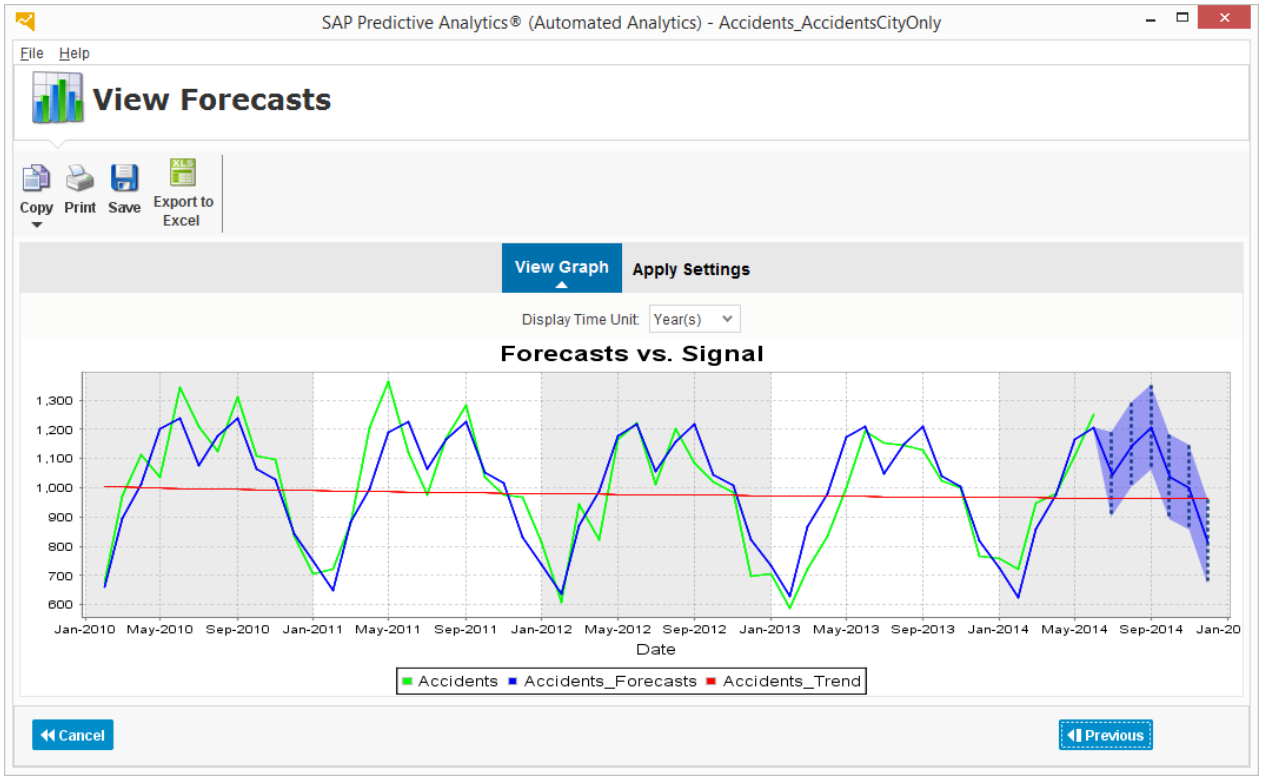
The lower the MAPE, the better obviously. Our MAPE of 0.057 means that on average (median measure) the forecasting error is 5.7%. So the model describes the historic data with 94.3% accuracy.



Click "Next" to start investigating and using the model.

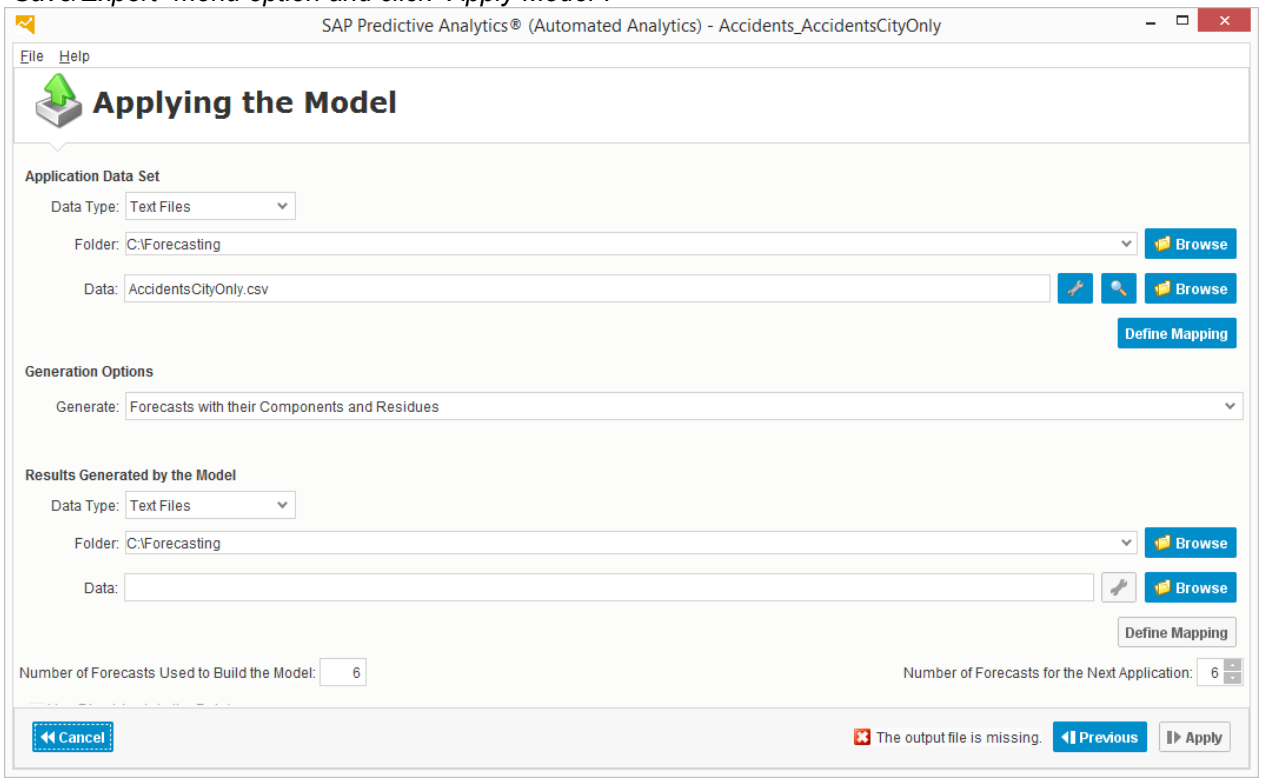


Click "View Forecasts" to see the historic data and the forecast.



The forecasts shown in blue capture the seasonality (less accidents in winter time) and the trend of less accidents happening over time.

To produce and save the forecast values for the next 6 months to file click "Previous", then select the "Save/Export" menu option and click "Apply Model".

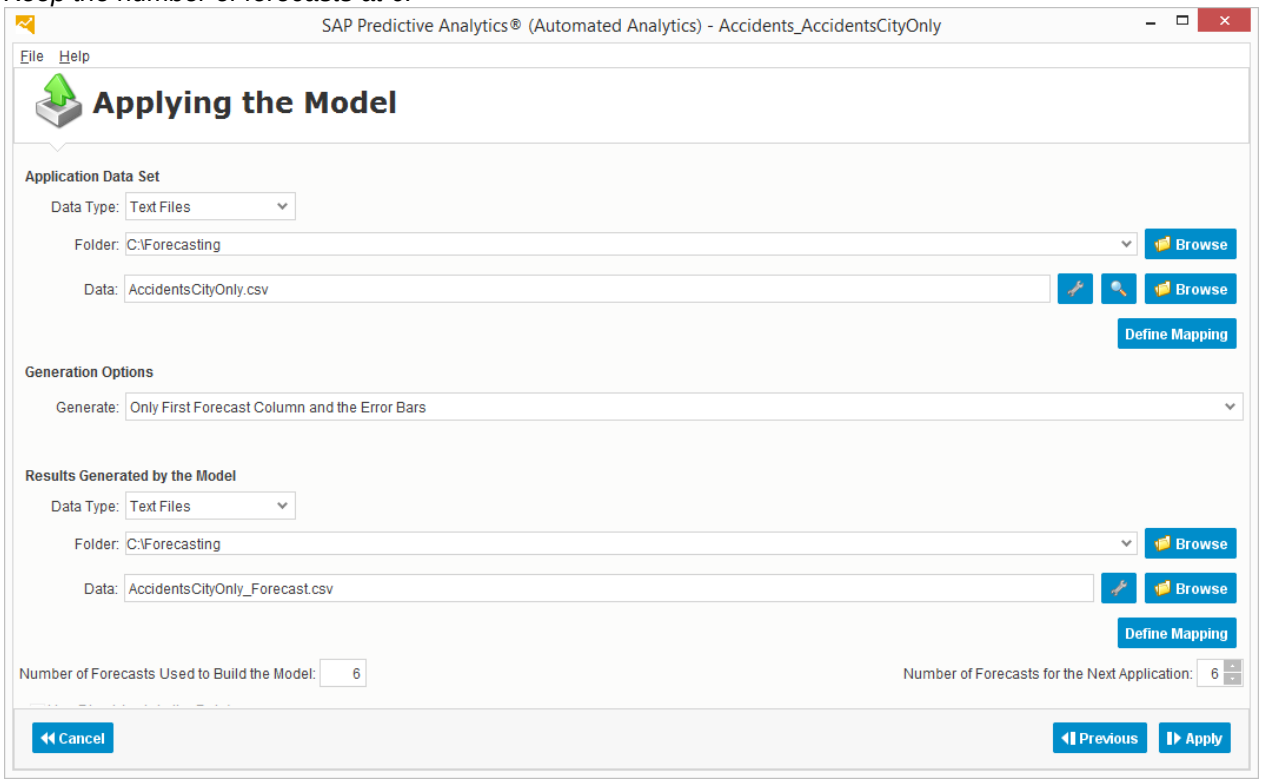


For the "Application Data Set" select the file we have already used to train the model. This should already be set by default.¨

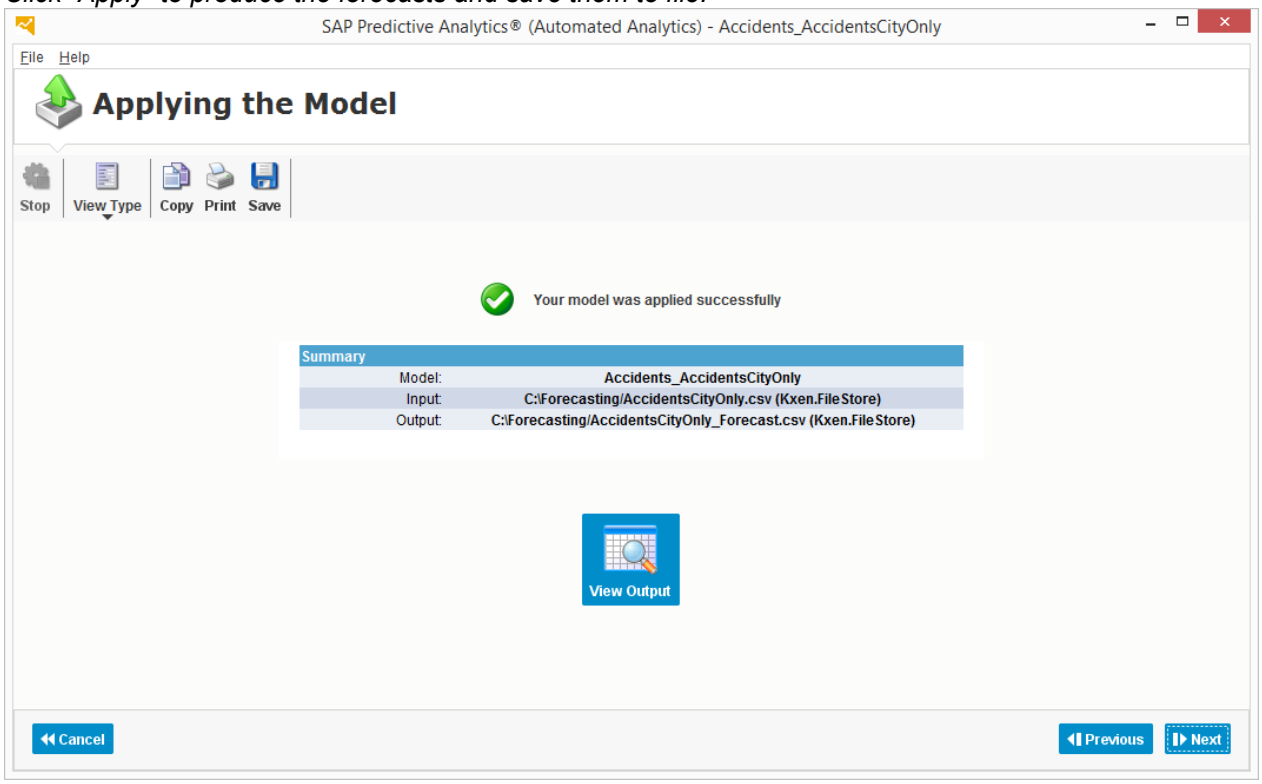
Set "Generate" to "Only First Forecast Column and the Error Bars" to keep it rather simple. This will output the predicted value for each date together with the error bars of the 95% confidence interval.

Specify in "Results Generated by the Model" to save the forecasts into a file called AccidentsCityOnly_Forecast.csv.

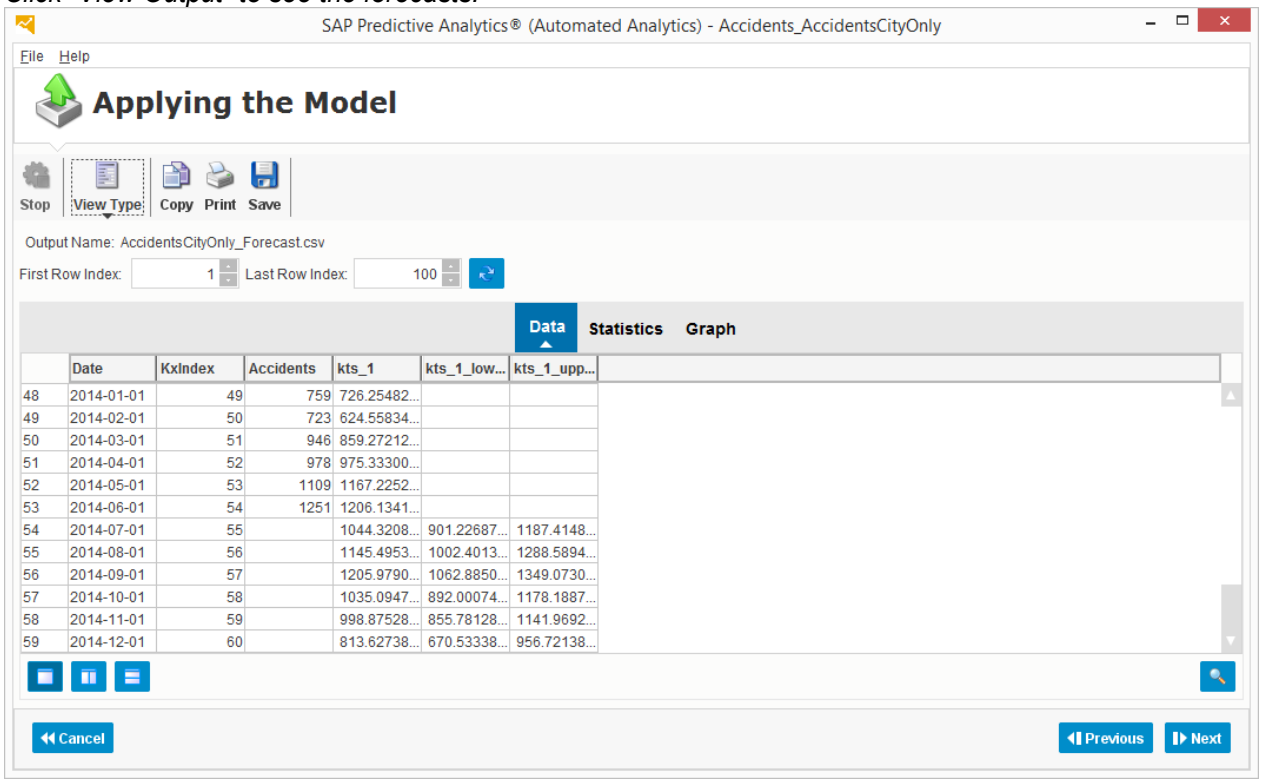
Keep the number of forecasts at 6.



Click "Apply" to produce the forecasts and save them to file.



Click "View Output" to see the forecasts.



	Date	KxIndex	Accidents	kts_1	kts_1_low...	kts_1_upp...
48	2014-01-01	49	759	726.25482...		
49	2014-02-01	50	723	624.55834...		
50	2014-03-01	51	946	859.27212...		
51	2014-04-01	52	978	975.33300...		
52	2014-05-01	53	1109	1167.2252...		
53	2014-06-01	54	1251	1206.1341...		
54	2014-07-01	55		1044.3208...	901.22687...	1187.4148...
55	2014-08-01	56		1145.4953...	1002.4013...	1288.5894...
56	2014-09-01	57		1205.9790...	1062.8850...	1349.0730...
57	2014-10-01	58		1035.0947...	892.00074...	1178.1887...
58	2014-11-01	59		998.87528...	855.78128...	1141.9692...
59	2014-12-01	60		813.62738...	670.53338...	956.72138...

The column "kts_1" holds the forecasts for the next 6 months. The same data is written into the file AccidentsCityOnly_Forecast.csv. Similarly the data could be written directly into a database for an end user or application to pick it up.

Dynamically forecasting multiple time series

Creating the forecast with the graphical user interface was easy. In case you have only a few time series to forecast, you will be done quickly. If however you have a much larger number of time series to forecast a different approach is needed. A retailer for instance that wants to forecast sales quantities on individual product items by stores has to repeatedly forecast thousands of time series.

In such a situation scripting is needed to produce the forecasts en masse. Everything that is done in the graphical user interface can be specified with a script. Through such scripts it is possible to forecast as many time series as needed. You do not have to write a new script yourself from scratch. Instead you carry out one forecast in the graphical interface and SAP Predictive Analytics provides you with the script that recreates what the user did in the graphical interface. Now you just need to modify the script to handle multiple time series and you can forecast as many time series as needed with fairly little effort.

Typically the script it split between two files. One script is training and saving the models. Another script loads these models and does the actual forecasts.

The following chapters apply this model to forecast three different time series through scripting. We forecast the number of accidents in the City, in the Countryside and on the Motorway (based on the data in AccidentsMultipleLocations.csv). The same principle applies to forecasting much larger number of time series.

This tutorial uses a flat file as data source. The same concept would apply connecting to a database. Either way, it is important that the dataset is sorted in descending order by date when it is filtered on one individual time series (so one single accident location in this example).

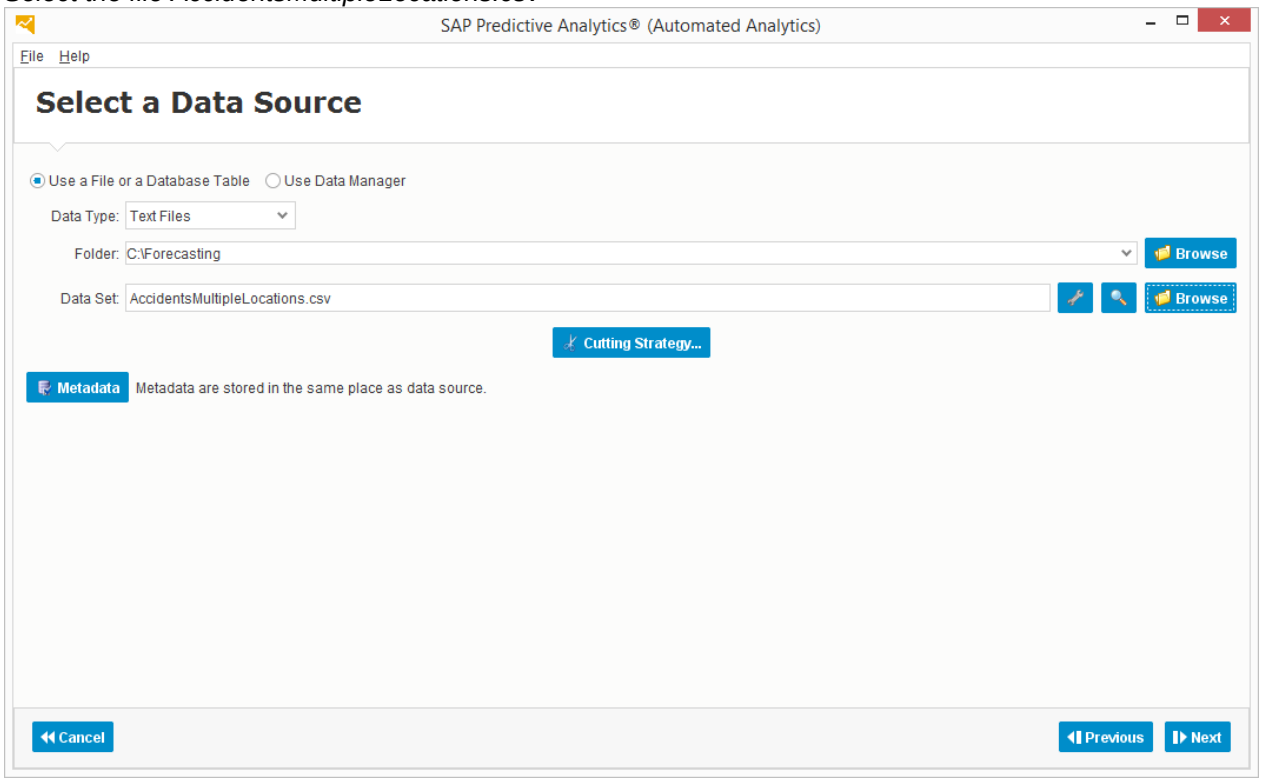
Obtaining script to train a model

We start by training our first model in the graphical user interface. As this forecast is very similar to above, the steps are described a little shorter.

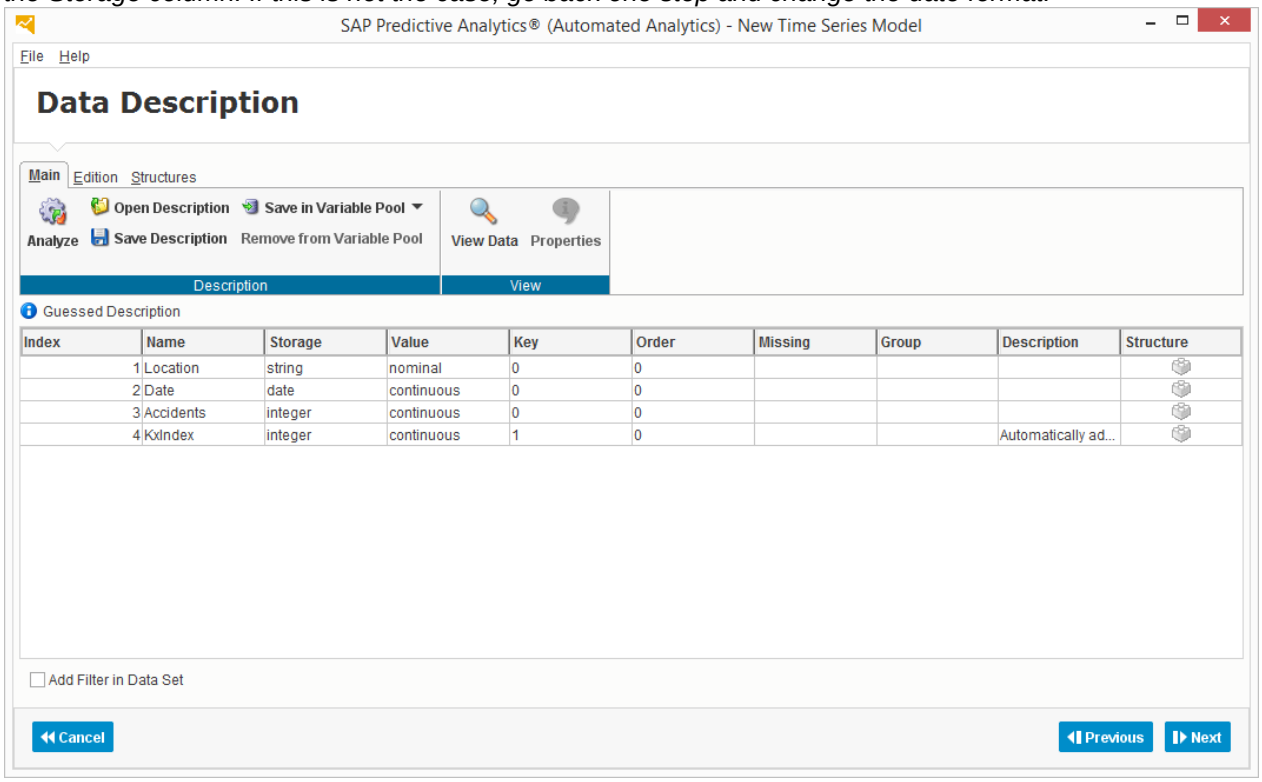
You may want to restart SAP Predictive Analytics before continuing so you know exactly which steps will be written into the script file.

From the main screen go into "Automated Analytics", then "Modeler" and select "Create a Time Series Analysis".

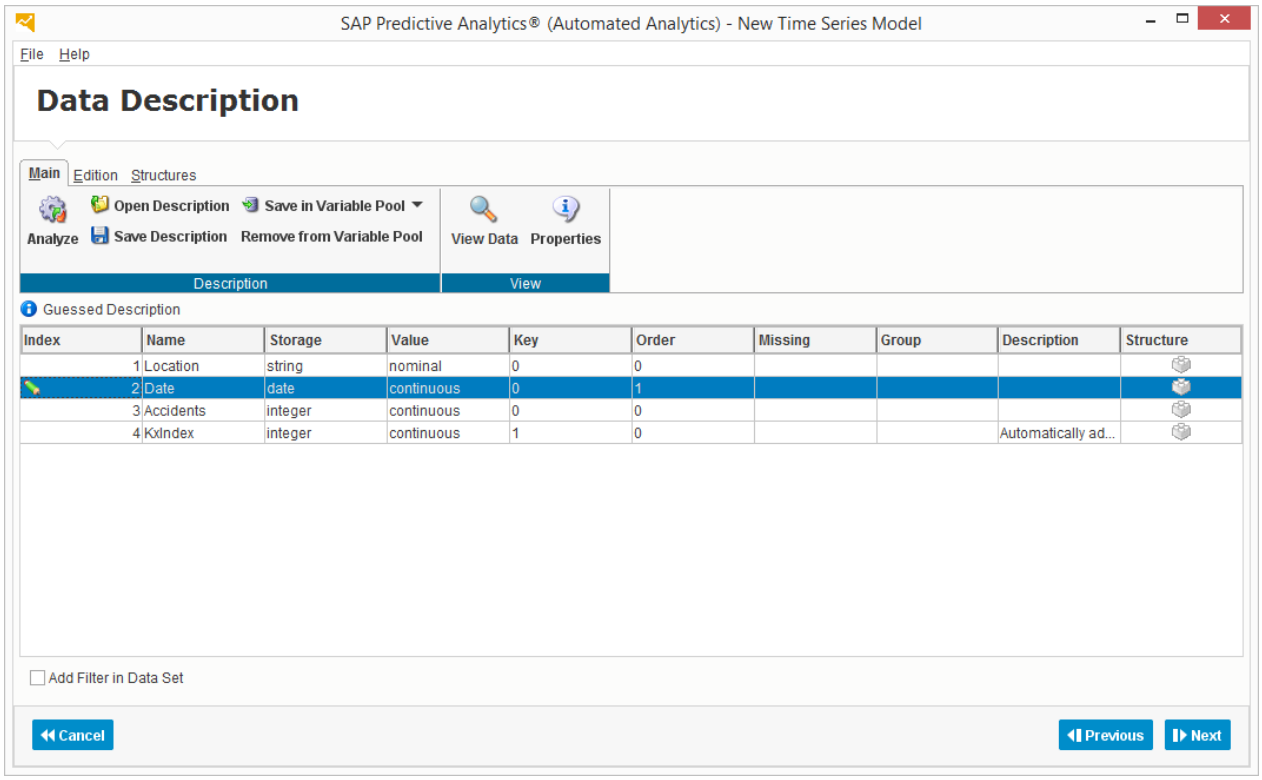
Select the file AccidentsMultipleLocations.csv



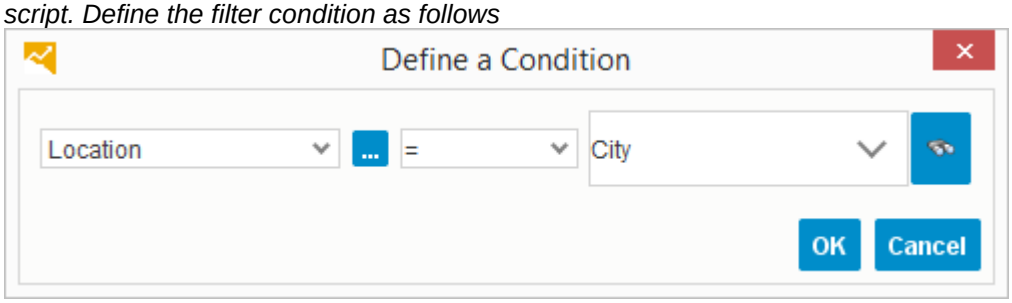
Continue and analyze the data structure. Remember that the date column has to be recognized as such in the Storage column. If this is not the case, go back one step and change the date format.



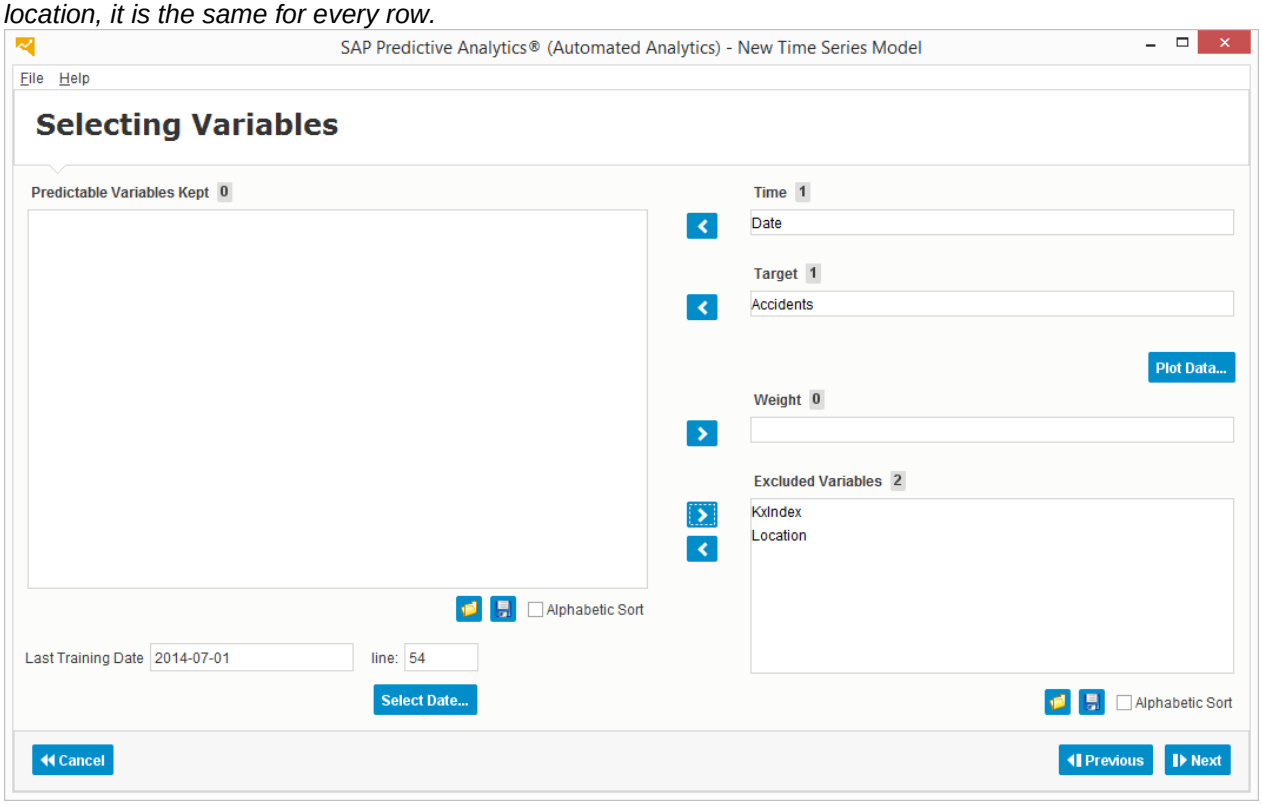
Set the order of the date column to 1, indicating that the individual time series in the data are sorted on this column.



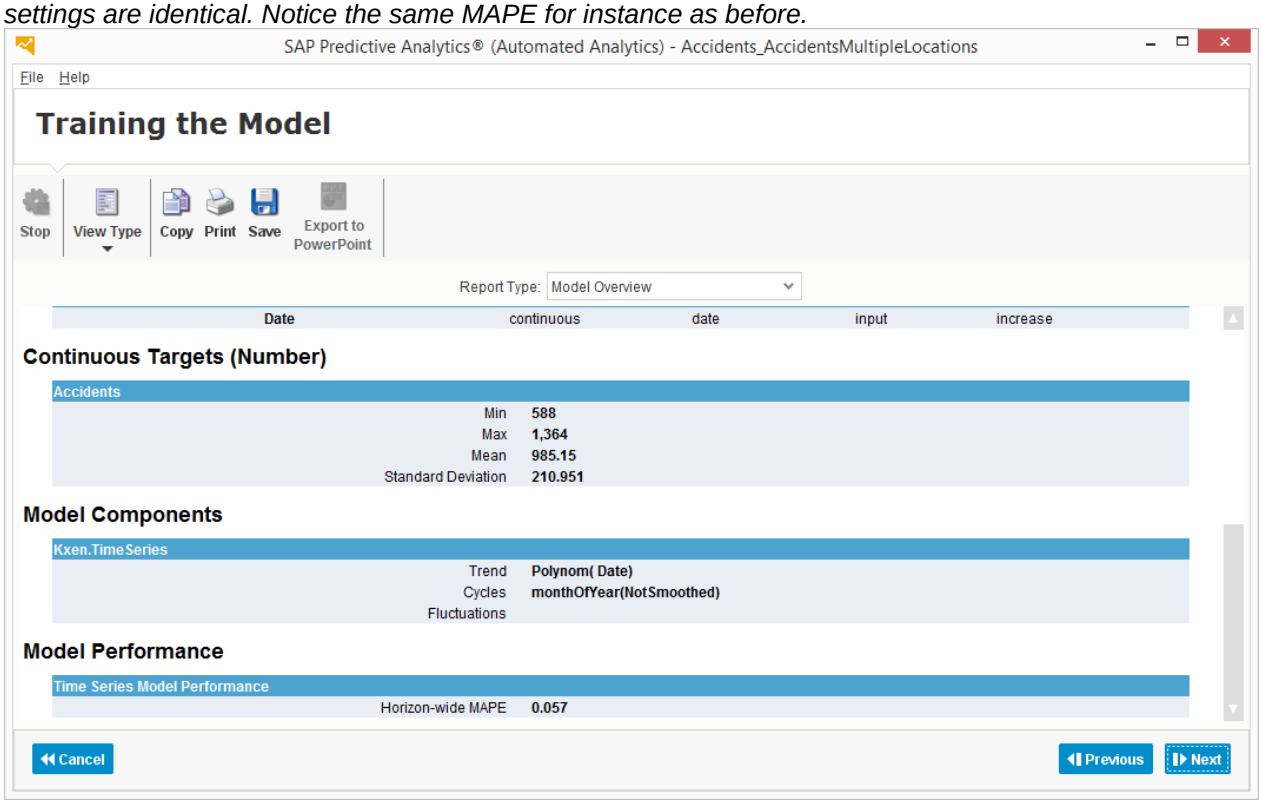
Tick the box "Add Filter in Data Set" and continue. Now filter the dataset on the first time series in our dataset by filtering on the location City. We do this by hand now and will later specify that more dynamically in the script. Define the filter condition as follows



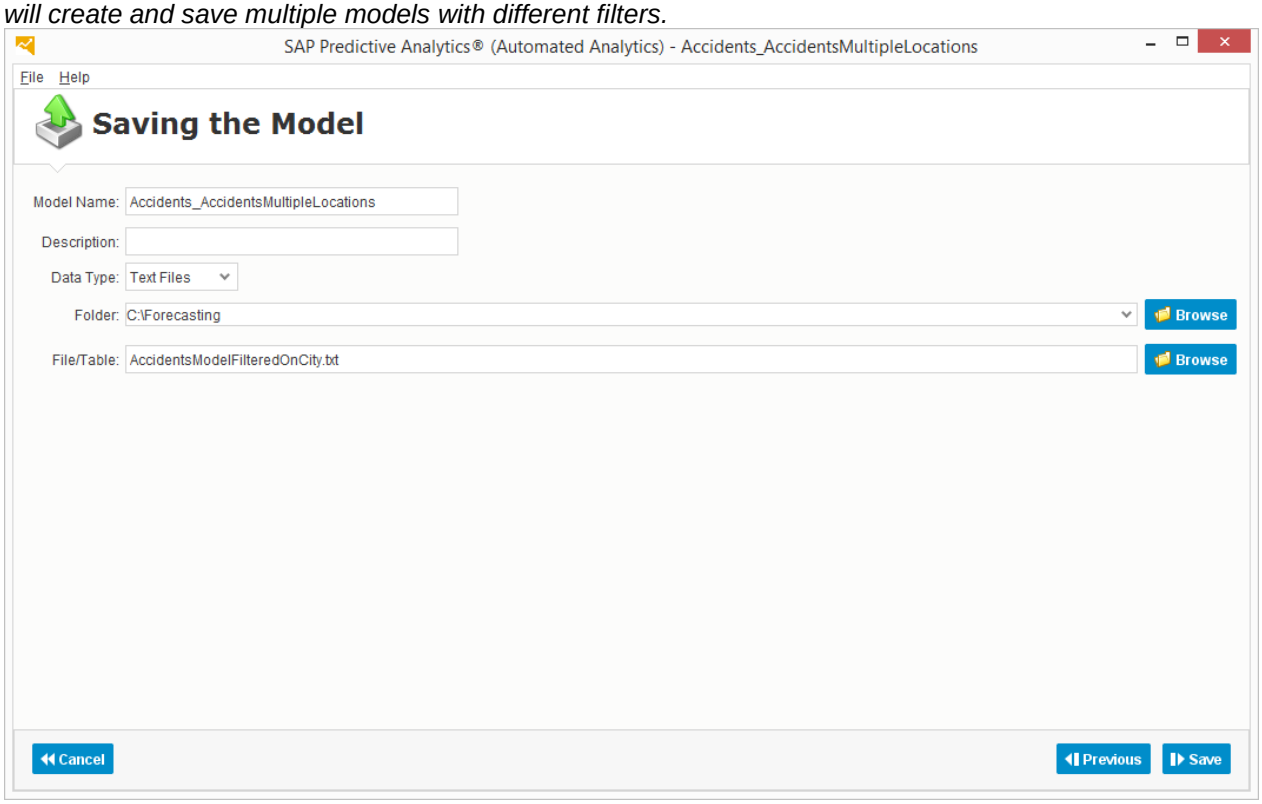
Click "OK and "Next". This brings you back to the familiar screen in which you can select the variables for the model. Exclude the location as it does not add any extra information to the model. Since we filtered on the location, it is the same for every row.



Continue with the forecast as before. Click "Next". Set the "Number of Forecasts" to 6 and hit "Generate". The "Model Overview" should show the same information as in the earlier chapter. The data and forecast settings are identical. Notice the same MAPE for instance as before.

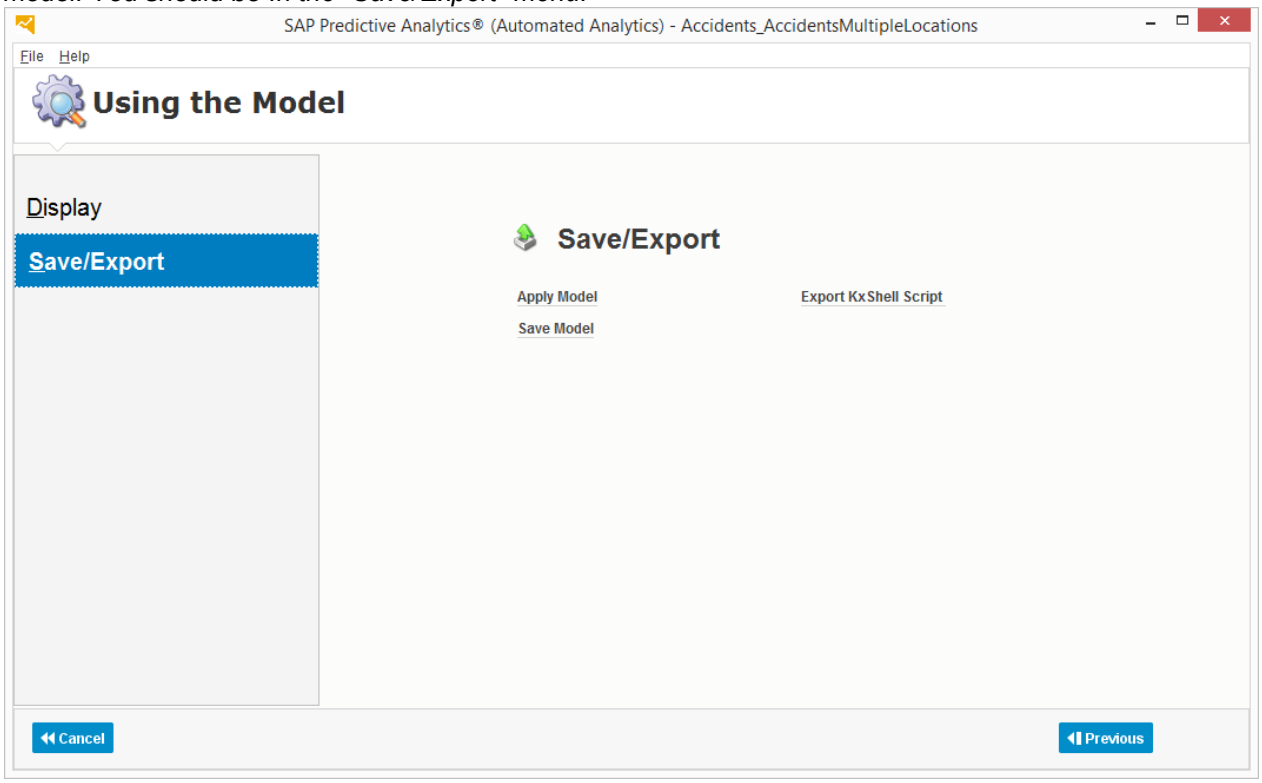


Continue with "Next". Select "Save/Export" and click "Save Model". Set the file name to AccidentsModelFilteredOnCity.txt. It is important that the filter value "City" is part of the name. Eventually we will create and save multiple models with different filters.

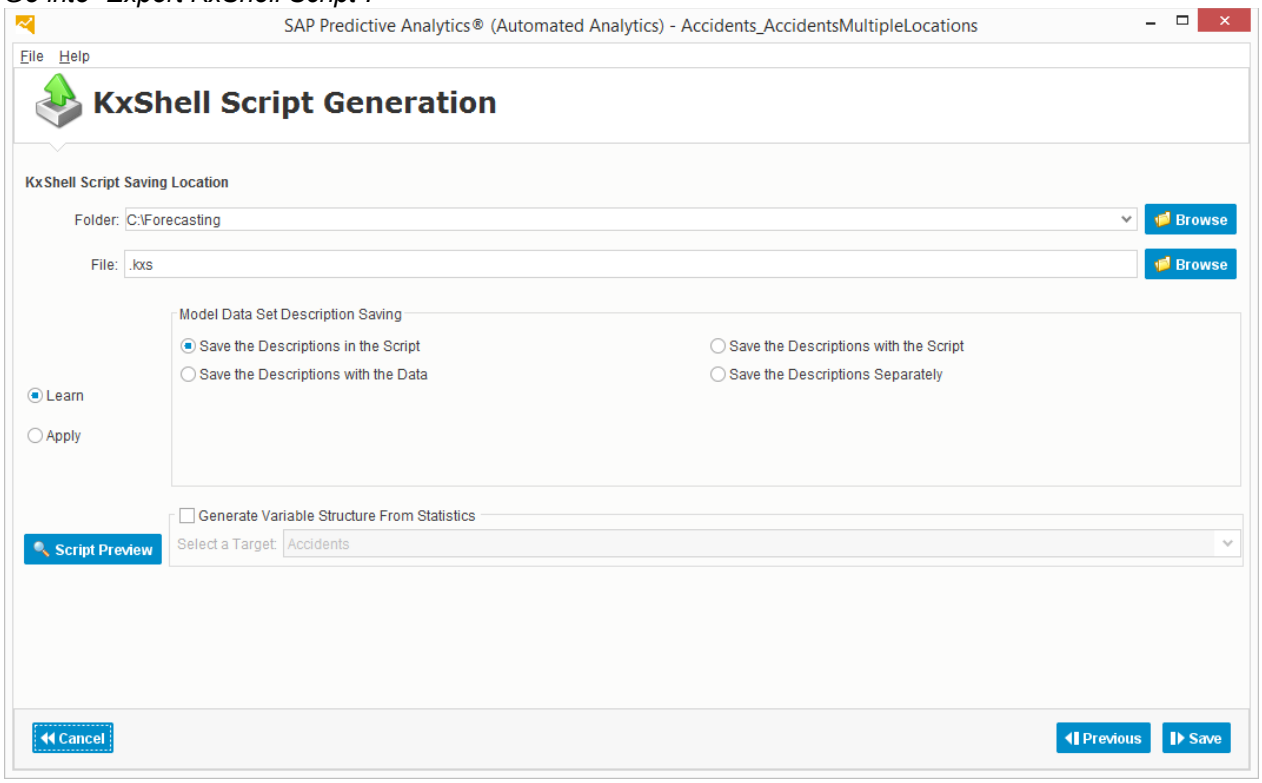


Click "Save" and the model should be created.

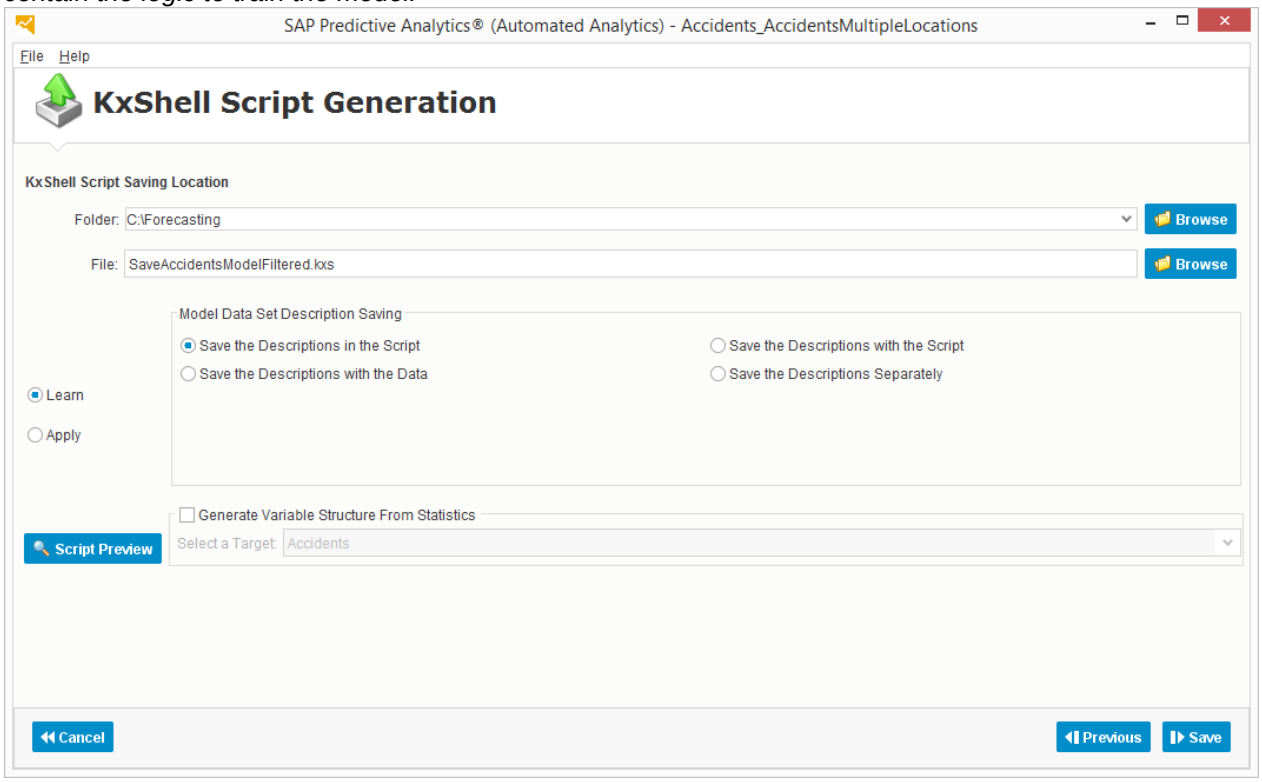
Now we have SAP Predictive Analytics produce the script that contains all the steps done until saving the model. You should be In the "Save/Export" menu.



Go into "Export KxShell Script".



Name the file "SaveAccidentsModelFiltered.kxs". Make sure "Learn" is selected. This ensures the script will contain the logic to train the model.

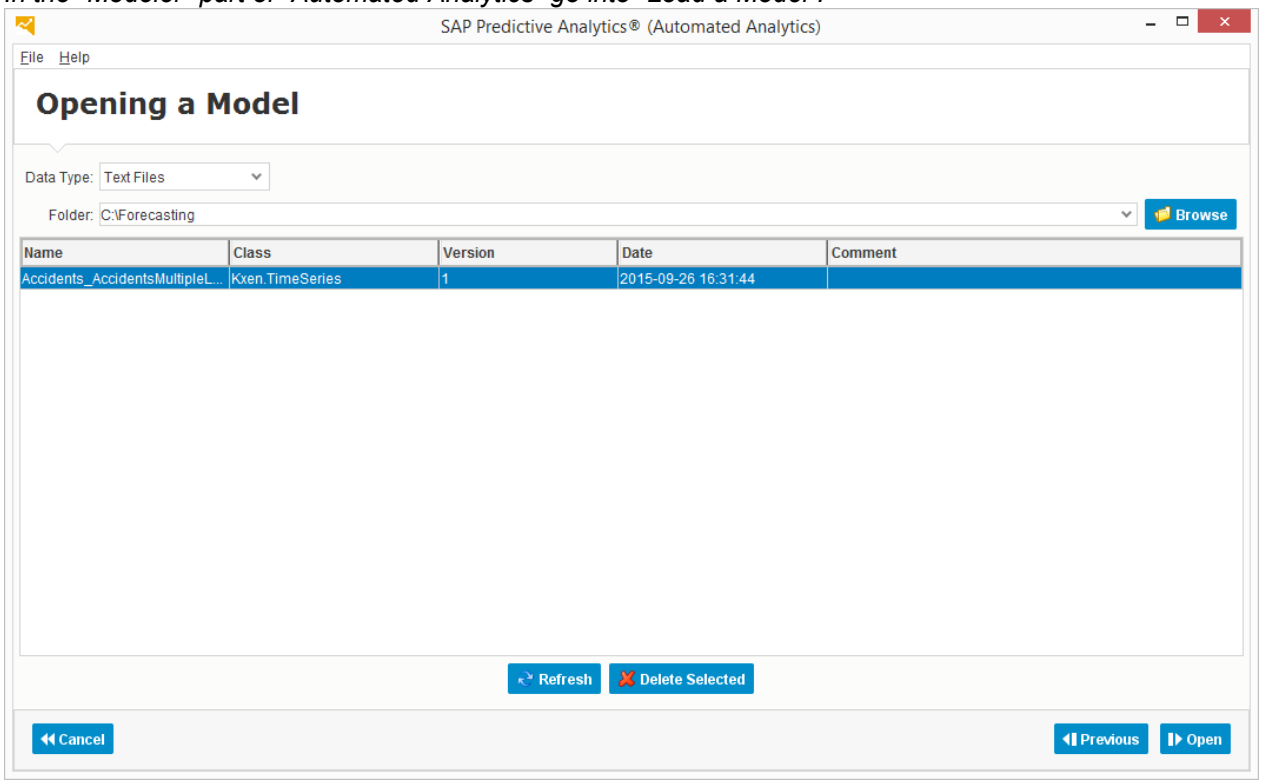


Click "Save" and the script file is produced.

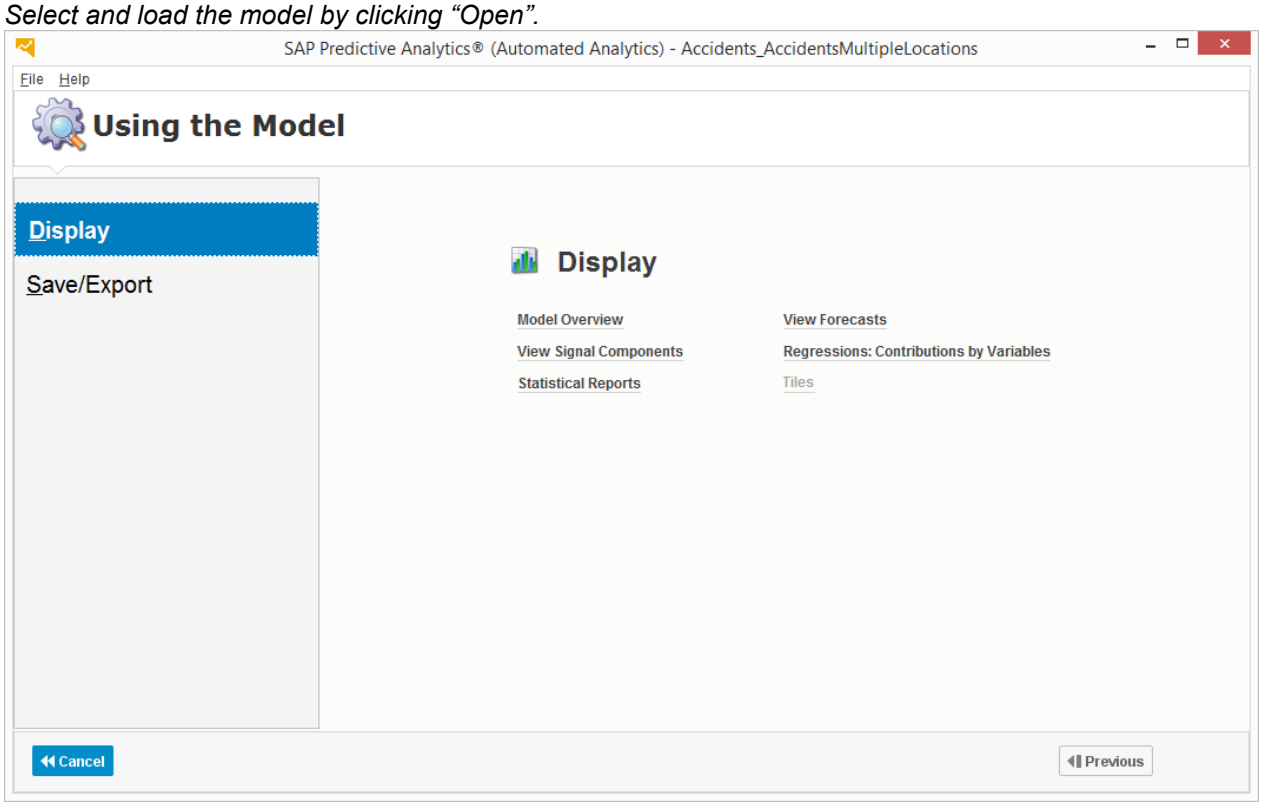
Obtaining script to apply a model

Before continuing, you may want to restart SAP Predictive Analytics again. Now in the graphical interface we load the model that we have just created with the data filtered on the City, we carry out the forecast and have the steps recorded again in a second script file.

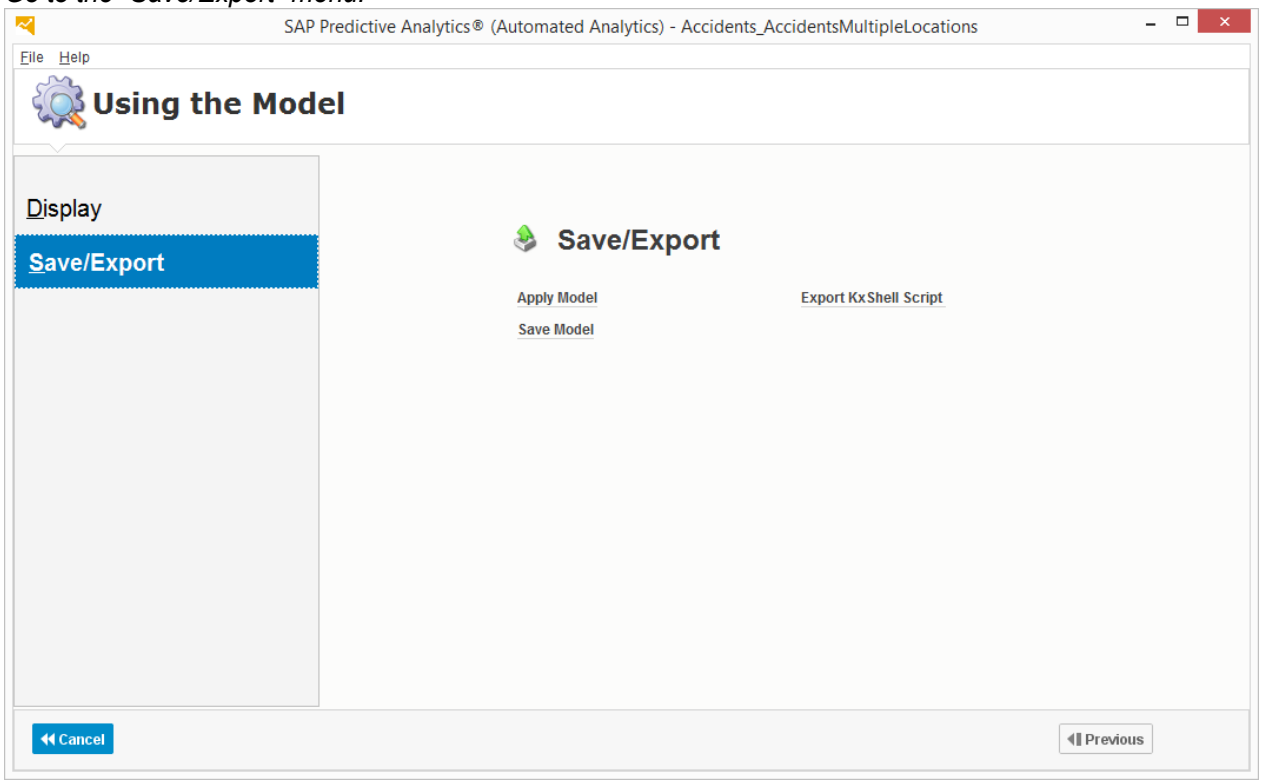
In the "Modeler" part of "Automated Analytics" go into "Load a Model".



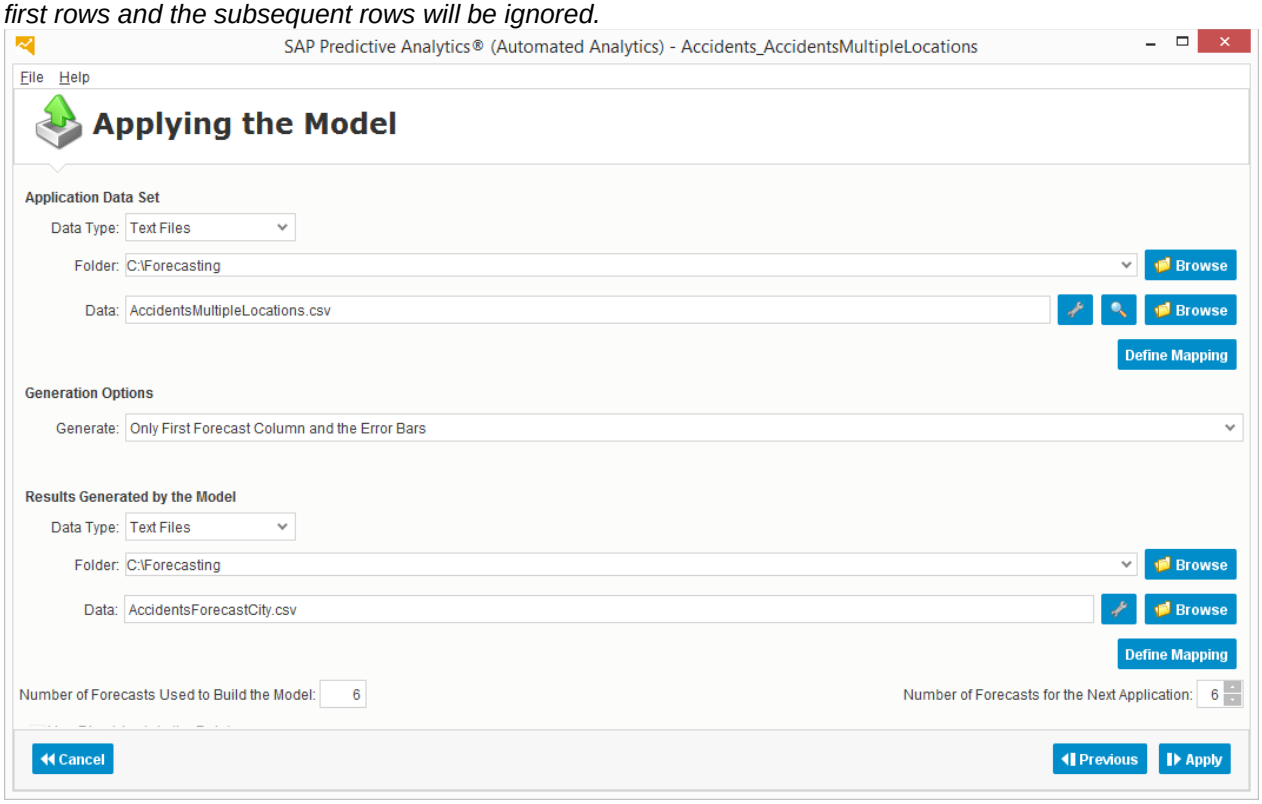
Select and load the model by clicking "Open".



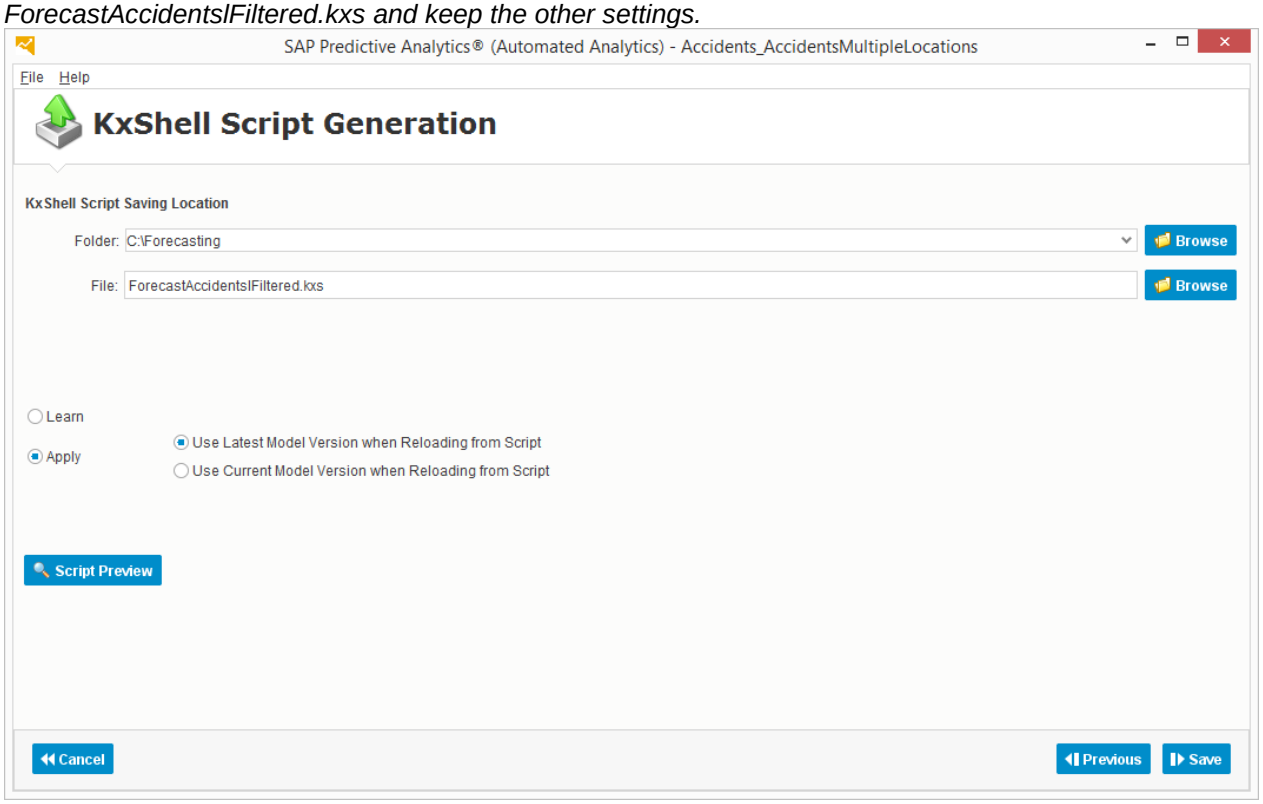
Go to the "Save/Export" menu.



Click "Apply Model". Keep most default settings, just set the "Generation Option" to "Only First Forecast Column and the Error Bars" and set the name of the output file to AccidentsForecastCity.csv. Even though the historic data contains locations other than City, we can use the file here as the City numbers are in the first rows and the subsequent rows will be ignored.



Click “Apply” and the forecast is created and saved. Continue with “Next” and as before go into “Export KxShell Script”. Now go into the “Apply” option, because we applied the trained model. Name the script ForecastAccidentsIFiltered.kxs and keep the other settings.



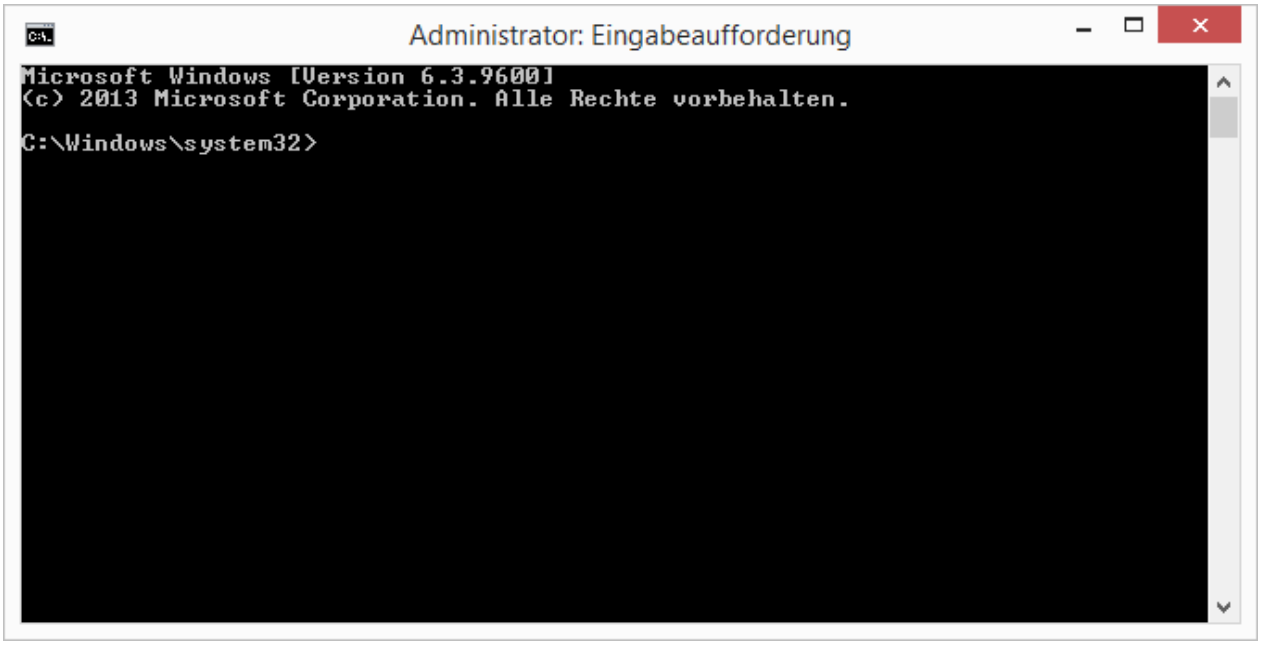
Click Save.

Single Forecast through script

Now that we have the two script files that can produce the forecast of a single time series. Try executing the scripts as they are before making them dynamic for multiple forecasts.

First tidy up the folder C:\Forecasting. Move *all files except the source data AccidentsMultipleLocations.csv and the two script files ending in .kxs into a new subfolder. We only need these three files to continue.*

Open a command prompt with Administrator rights. Find the file cmd.exe, right-click on it and select "Run as administrator".



Change the path as follows. Depending on your version of SAP Predictive Analytics the path might be different. In that case use the path that contains the file KxShell.exe.
cd C:\Program Files\SAP Predictive Analytics\Desktop 2.3\Automated\EXE\Clients\CPP

Call the script to train the model (change the file path to where you would like to have it on your computer):
kxshell C:\Forecasting\SaveAccidentsModelFiltered.kxs -DMODEL_SAVE_STORE_NAME=C:\Forecasting\

Notice how the model file has been created. Now call the second script to load that model and to forecast.
kxshell C:\Forecasting\ForecastAccidentsIFiltered.kxs

The forecast has been written to file. You have executed a forecast through script!

Multiple Forecasts through script

Now that we know how to create and execute scripts to forecast a single time series, we modify the scripts to be able to forecast multiple time series at once. Scripts are made dynamic with parameters that are passed through the command line and used inside the script.

To forecast different time series by location in our example, we modify the scripts so that the desired location is passed through the command line. So for each time series we execute the scripts with the desired filter value.

The script that trains the models filters the data on this value and saves the produced model with the location in the name, not to overwrite other models. The script that triggers the forecast needs to be passed the same filter value to open the corresponding trained model, to filter the dataset and to save the forecast as a separate file, again with the filter value in its name.

Such a command line parameter is passed as argument to the kxshell, prefixed by "–D" (without the quotes). You have already done this in fact, see the paragraph "Single Forecast through script". You passed the file location as follows: -DMODEL_SAVE_STORE_NAME=C:\Forecasting\

In the script, the value is retrieved easily. Just add a $-sign in front of the parameter name and the placeholder will be replace with the value passed through command line.

Training the Models

In our example we call the parameter MYLOCATION. Edit the file SaveAccidentsModelFiltered.kxs in Notepad or any other editor you prefer.

1) To add the parameter value to the model and model file names, change these lines:
default MODEL_SAVE_NAME "Accidents_AccidentsMultipleLocations"
default MODEL_SAVE_SPACE "Accidents_AccidentsMultipleLocations.kxen"
default MODEL_SAVE_COMMENT "The model 'Accidents_AccidentsMultipleLocations"

to this:
default MODEL_SAVE_NAME "Accidents_AccidentsMultipleLocations_$MYLOCATION"
default MODEL_SAVE_SPACE "Accidents_AccidentsMultipleLocations_$MYLOCATION.kxen"
default MODEL_SAVE_COMMENT "The model 'Accidents_AccidentsMultipleLocations_$MYLOCATION' has been saved"

2) To filter the dataset, change this line
s.changeParameter "Parameters/FilterCondition/SimpleFilter1/Value" "City"

to this:
s.changeParameter "Parameters/FilterCondition/SimpleFilter1/Value" $MYLOCATION

3) To update the status message change this line
print Model Accidents_AccidentsMultipleLocations has been saved.

to
print $MODEL_SAVE_COMMENT

4) Save the file and call it with these three commands
kxshell C:\Forecasting\SaveAccidentsModelFiltered.kxs -DMODEL_SAVE_STORE_NAME=C:\Forecasting\ -DMYLOCATION=City
kxshell C:\Forecasting\SaveAccidentsModelFiltered.kxs -DMODEL_SAVE_STORE_NAME=C:\Forecasting\ -DMYLOCATION=Countryside
kxshell C:\Forecasting\SaveAccidentsModelFiltered.kxs -DMODEL_SAVE_STORE_NAME=C:\Forecasting\ -DMYLOCATION=Motorway

We now have three different, individually trained models.

Forecasting the Multiple Time Series
So to use these multiple models, we need to modify the script that does the forecasts. Edit the file ForecastAccidentslFiltered.kxs

1) To add the parameter value to the model file name that will be loaded, change this line:
default MODELNAME "Accidents_AccidentsMultipleLocations"

to
default MODELNAME "Accidents_AccidentsMultipleLocations_$MYLOCATION"

2) To add the parameter name to the file name of the forecasts, change this line
default APPLYOUT_SPACE "AccidentsForecastCity.csv"

to
default APPLYOUT_SPACE "AccidentsForecast_$MYLOCATION.csv"

3) Filtering the dataset on our individual time series is a little more complex as we did not have this option in the graphical interface. Hence the script does not include any filtering that can be modified. However, the script to train the model includes this logic. It only needs a small change. See this code.

```
# Build space filter tree structure
bind m DataSet Training s
s.getParameter ""
s.bindParameter "Parameters/FilterCondition" FilterCondition1
FilterCondition1.insert "SimpleFilter1" SimpleFilter2
delete SimpleFilter2
delete FilterCondition1
s.validateParameter
s.getParameter ""
# Fill the space filter parameters
s.changeParameter "Parameters/FilterCondition/Operator" "And"
s.changeParameter "Parameters/FilterCondition/SimpleFilter1/Operator" "Equal"
s.changeParameter "Parameters/FilterCondition/SimpleFilter1/Variable" "Location"
s.changeParameter "Parameters/FilterCondition/SimpleFilter1/Value" $MYLOCATION
s.validateParameter
delete s
```

You do not have to type this by hand! Look into the file SaveAccidentsModelFiltered.kxs, where you find this code. Copy these lines and paste them into ForecastAccidentslFiltered.kxs right after the line
delete t

Finally change this line
bind m DataSet Training s

to
bind m DataSet ApplyIn s

4) Save the file and call it with these three commands
kxshell C:\Forecasting\ForecastAccidentslFiltered.kxs -DMYLOCATION=City
kxshell C:\Forecasting\ForecastAccidentslFiltered.kxs -DMYLOCATION=Countryside
kxshell C:\Forecasting\ForecastAccidentslFiltered.kxs -DMYLOCATION=Motorway

You have now created three individual forecasts! They are stored in AccidentsForecast_City.csv, AccidentsForecast_Countryside.csv and AccidentsForecast_Motorway.csv.

The concept could be further automated, for instance by reading the filter values from the database and by dynamically creating individual models and forecasts for as many filter values as found.

You are now able to produce as many individual time series as required!

HINTS AND TIPS / MORE INFORMATION

Remember that:

- the individual time series, that will be forecasted, needs to be sorted by date in descending order.
- in the “Data Description” the Storage type of the date columns has to be “date”.
- in the “Data Description” the Order of the date columns has to be flagged with the value 1.
- the training dataset must include the dates and predictor variables for the dates you want to forecast. (not needed in case there are no additional predictor variables).

For further information see the help file “Time Series Scenarios” on http://help.sap.com/pa